Lär dig

VIC 20

En självinstruerande handbok

commodore COMPUTER

© Copyright HANDIC Electronic AB 1981

Materialet får inte helt eller delvis kopieras i någon form utan skriftlig tillåtelse av HANDIC Electronic AB

Lär dig VIC 20

En självinstruerande handbok

Handic Electronic AB

FÖRORD

I denna manual skall du få lära dig hur du behärskar din VIC. Du kommer att upptäcka att datorer inte är så dumma som du tror... de är ännu dummare. Din VIC kommer nämligen att göra precis som du säger till den.

Om det är den första gång du bekantar dig med en dator så är denna handbok en utmärkt start. Till skillnad från alla andra handböcker så behöver du bara läsa det första kapitlet (LÄR KÄNNA DIN VIC 20). När du gjort det kan du utan vidare gå till vilket kapitel som du vill. Om du är intresserad av musik läser du kapitel 5.

Första sidan på varje kapitel innehåller ett programexempel för att du skall få ett grepp om vad det handlar om. Skriv bara in programmet precis som det står i handboken och se vad som händer. Resten av kapitlet förklarar vad programmet gjorde och visar hur du kan utöka programmet. Kapitel 7 förklarar en del viktiga programmeringstekniker och förklarar hur dessa används i programexemplen.

Är du en erfaren programmerare kan du använda VIC som vilken dator som helst för att göra avancerade program på. Om du tidigare arbetat med något av COMMODORE's datorer underlättar det mycket eftersom BASIC och grafiksymbolerna är i stort sett desamma som används PET/CBM. Mer ingående information om programmering finns i bilagorna. För mera sofistikerad programmeringsinformation se VIC PROGRAMMERINGSHANDBOK, som du kan köpa hos din närmaste VIC-återförsäljare.

Om du inte har ett brinnande datorintresse så bör du titta på vad som finns i VIC's ständigt växande bibliotek av pluginkort och programkassetter. VIC's pluginkort stoppas in på VIC's baksida och startas automatiskt. Färdiga program finns också att köpa på kassetter som skall användas tillsammans med COMMODORE's kassettenhet C2N.

De program som finns är spelprogram typ INVANDERS, utbildningsprogram som lär dig eller dina barn att skriva eller räkna samt olika program för att lösa olika räkneproblem.

Bland de tillbehör som finns till VIC kan nämnas skrivare, kassettbandspelare och skivminne. (Se bilaga A).

Datorer är på väg att bli en ännu viktigare del av vårt dagliga liv, i hemmet, i skola eller på arbetet. De som lär sig att hantera datorer i dag kommer att ha en stor fördel i den kommande framtiden. VIC är inte bara en hjälp för dig att lära känna datorer utan den har också möjligheten att vara flexibel och utbyggbar för att kunna hänga med i dagens snabba expansionstakt.

Mycket nöje!

Innehåll

	Förord	II
	Uppkoppling av VIC 20	V
1	Lär känna din VIC 20	1
1.1	Att komma i gång	3
1.2	Ditt första datorprogram	7
2	Bildskärm och tangentbord	11
2.1	Ditt första grafiska tecken	14
2.2	En beskrivning av tangentbordet	17
2.3	Att skriva på skärmen	21
2.4	VIC som miniräknare	24
2.5	Introduktion till färg	25
3	Färg och grafik	27
3.1	Programmering av färg	30
3.2	Färgtangenter	32
3.3	Bakgrunds och ramfärger	34
3.4	Kombinationer av bakgrunds och ramfärger	37
3.5	Färg till tecken	37
3.6	Teckenpositioner	39
3.7	Slumpfärger	40
3.8	Ljud och färg	45
3.9	Grafik på tangentbordet	47
3.10	Grafik i överskrifter och rubriker	48
4	Figur ritning	51
4.1	Flygande fåglar	53
4.2	Studsande bollar	57
4.3	Kontroll av markör	60
4.4	POKE och PEEK	61
5	Ljud och musik	67
5.1	Att göra musik	69
5.2	VIC's fyra röster	71
5.3	Brusgenerator	74
5.4	VIC som ett piano	76
5.5	Spela melodier	78
5.6	Några ord om POKE	80

6 Prata med din VIC 81
6.1 Vad heter du 83
6.2 Variabler 86
6.3 Välj en ton 88
6.4 GET-kommandot 89

7 Introduktion till programmering 93
7.1 Ditt första BASIC-kommando 95
7.2 Slumptal 103

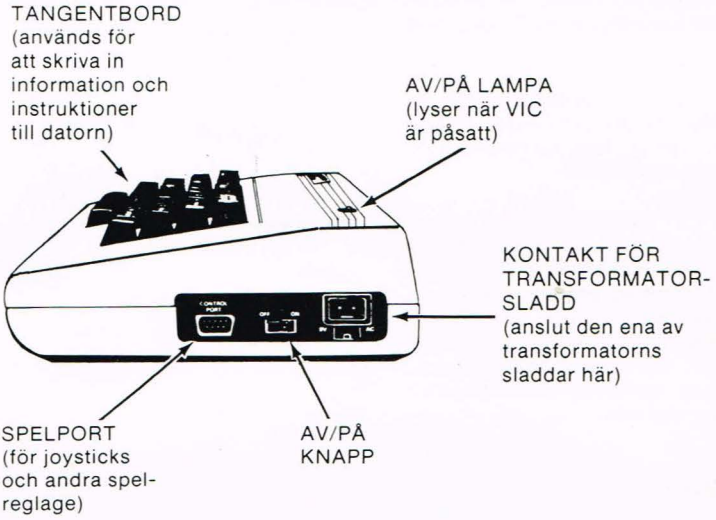
A. VIC systemtillbehör 106
B. Kassettstation 109
C. VIC BASIC 113
D. Kommando förkortningar 133
E. Kombinationer av bakgrunds- och ramfärger 134
F. Musiktoner 135
G. Ljudeffekter 136
H. Bildskärmskoder 139
I. Minnespositioner för bildskärm 143
J. ASCII koder 145
K. Matematiska funktioner 148
L. Anslutningskontakter 150
M. VIC program 153
N. Felmeddelanden 160

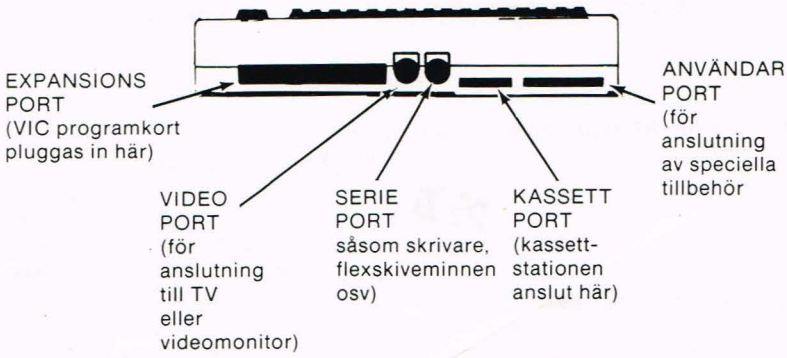
Register 162

UPPKOPPLING

Följande steg för steg beskrivning visar hur du kopplar upp din VIC till en TV, och kontrollerar att den fungerar riktigt.

Låt oss först ta en titt på VIC 20:





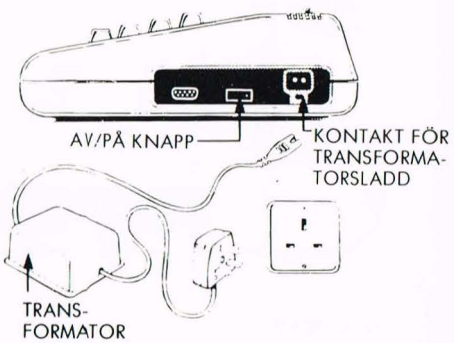
1 Kontrollera ditt VIC-paket, det ska innehålla följande produkter:
- VIC 20 datorenhet
- Transformator (en låda med två sladdar på)
- RF Modulator (liten metallåda med en kort kabel på)
- Videokabel

2 Du behöver två stycken nätuttag, ett för VIC och ett för din TV.

3 Placera din VIC och TV så att du kan skriva bekvämt på tangentbordet och samtidigt se på TV-skärmen.

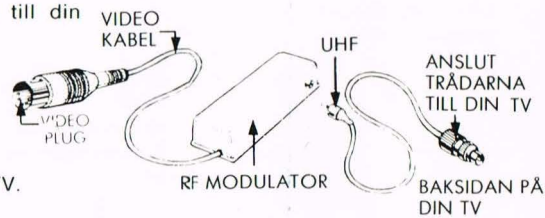
4 Lokalisera AV/PÅ knappen på VIC's högra sida. Försäkra dig att den är i läge AV (intryckt vid markering OFF).

5 Det kommer två sladdar ur transformatorn. Anslut nätsladden i ett nätuttag, och den andra sladden i kontakten på VIC's högra sida. OBS transformatorn står på även om VIC inte används, drag därför ut nätsladden till transformatorn när VIC inte används.



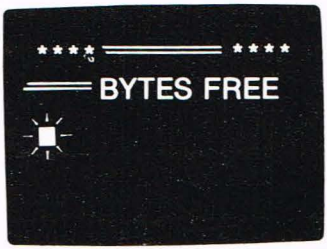
6 Anslut videokabeln till VIC's baksida och till RF-modulatorlådan. Se till så att du monterar kabeln på VIDEO-porten och inte på SERIE-porten till höger om den.

7 Anslut RF-modulatorn till din TV...



8 Anslut och sätt på din TV.

9 Sätt på VIC (AV/PÅ lampan kommer att lysa). Om inte lampan lyser se på felsökningsbladet för att hitta orsaken.



10 Detta är vad du borde se på skärmen. Det tar ett par sekunder innan bilden kommer fram. Får du inte fram bilden, stäng av VIC och vänta i ca 5 sekunder och försök sedan igen.

11 Justera färgerna, om det behövs, med hjälp av färgkontrollerna på TV'n. Färgerna beror mycket på kvaliteten och åldern på TV-apparaten.

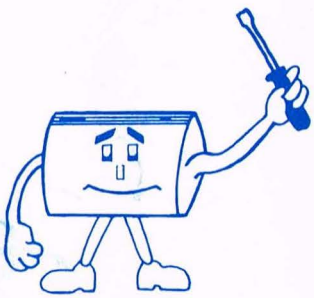
12 Får du problem med något av dessa steg, gå igenom felsökningsbladet för att hitta orsaken.

NU... kan du börja använda din VIC.

OBS: En videomonitor kan användas istället för en TV, då behövs inte RF-modulatorn. Även en svart/vit TV kan användas, färgerna blir en gråton skala istället.

FELSÖKNINGSSCHEMA

SYMPTOM	ORSAK	ÅTGÄRD
INGEN BILD (LAMPAN LYSER EJ)	VIC inte på	Försäkra dig om att strömbrytaren är i läge ON
	VIC inte ansluten	Kontrollera att sladden sitter i kontakten
	Transformator inte ansluten	Kontrollera anslutningen vid vägguttag
	Dålig säkring i VIC	Lämna in din VIC till en auktoriserad återförsäljare för utbyte av säkring
INGEN BILD (LAMPAN LYSER)	(Försök med att stänga av VIC, vänta i några sekunder, och slå sedan på VIC igen)	
	TV:n inställd på fel kanal	Kontrollera om det finns någon bild omkring kanal 30
	Fel anslutning	VIC anslutes till UHF-ingången
	Modulator inte ansluten	Kontrollera anslutning till den 5-poliga video-kontakten
BILD UTAN FÄRG	Dåligt inställd TV	Justera TV:n
BILD MED DÅLIGA FÄRGER	Dålig justering av färg-kontroller på TV:n	Justera färgkontrollerna på TV:n
BILD MED BAKGRUNDSLJUD	TV-volymen för hög	Justera volymkontrollen
BILD BRA MEN INGET LJUD	TV-volymen för låg	Justera volymkontrollen



*VIC använder en 3A trög säkring.

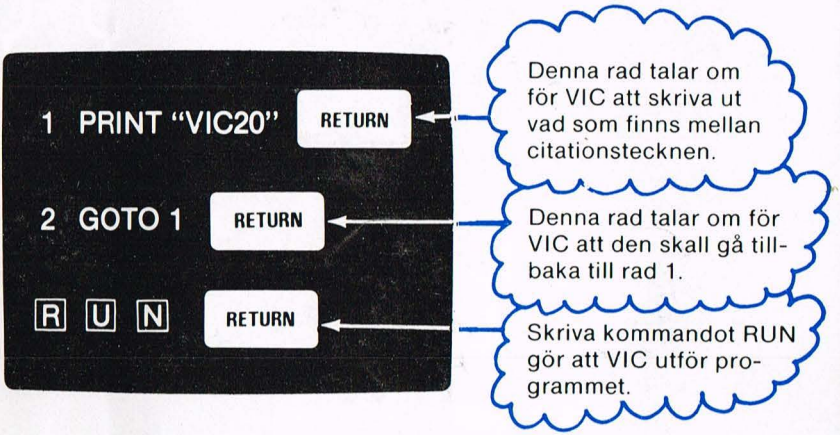
1

Lär känna din VIC 20

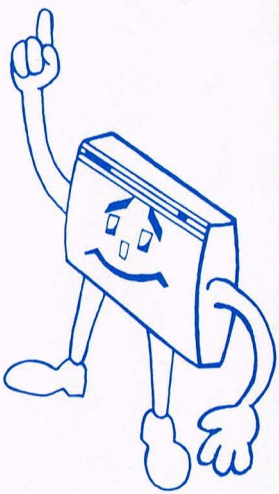
- Att komma igång
- Ditt första datorprogram

Skriv detta program

Skriv programmet precis som rutan nedan visar, och se vad som händer!



För att stoppa programmet, tryck RUN STOP tangenten.



Här följer en introduktion

När du har slagit på din VIC, väntar den på att du skall tala om vad den skall göra. Den mörkblå blinkande rektangeln kallas för markör, och är en signal från VIC att den väntar på dig.

TIPS:

Har du skrivit ett tecken på skärmen som du inte vill skall vara där, tryck på

INST DEL tangenten.

Denna tangent kommer att radera tecknet direkt till vänster om markören.

Använd denna tangent när du vill radera oönskade tecken.

Låt oss börja med ett experiment. Tryck på följande tangenter:

P R I N T

Som du ser flyttar sig markören ett steg till höger för varje tangenttryckning. Markören indikerar positionen för var nästa tecken skall placeras. Leta rätt på denna tangent:

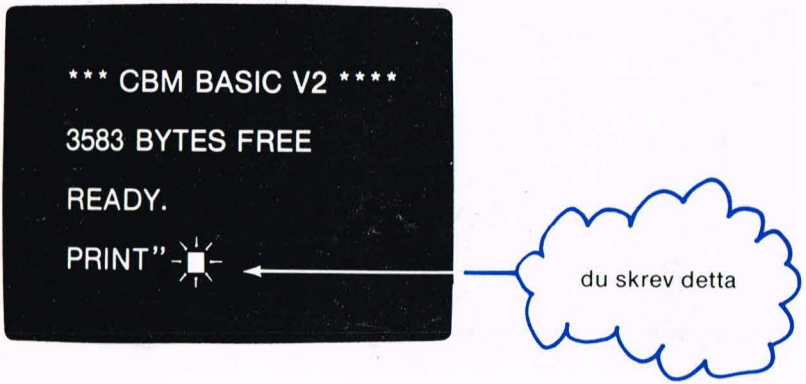
SHIFT

Det finns två stycken som har samma funktion.

Håll SHIFT tangenten nedtryckt, och tryck på tangenten: 2"

Efter att ha tryckt på SHIFT tangenten kan du släppa upp den.

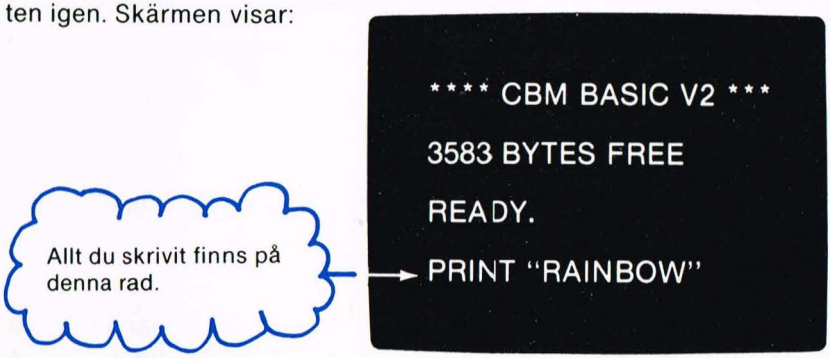
Skärmen ser ut som följer:



När 2" tangenten var nedtryckt samtidigt som SHIFT tangenten, blev det ett citationstecken på skärmen.* Låt oss fortsätta. Tryck på följande tangenter:

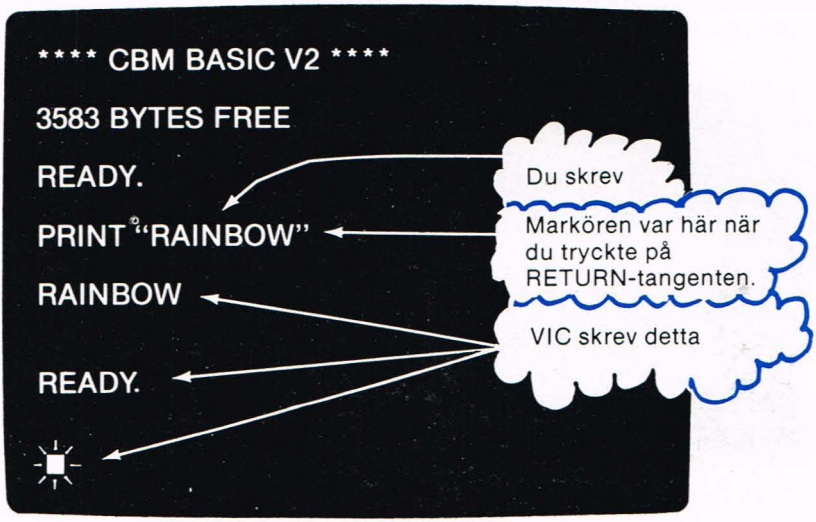
R A I N B O W

Tryck slutligen ned SHIFT tangenten och tryck på tangen ten igen. Skärmen visar:



* Om siffran 2 kom fram på skärmen istället för " tecknet, var inte SHIFT tangenten nedtryckt. Tryck på INST DEL tangenten för att radera 2, och försök igen.

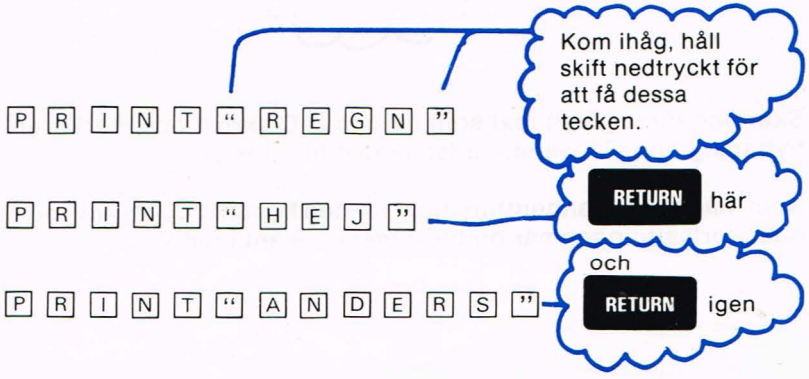
Tryck på RETURN tangenten och se på skärmen.



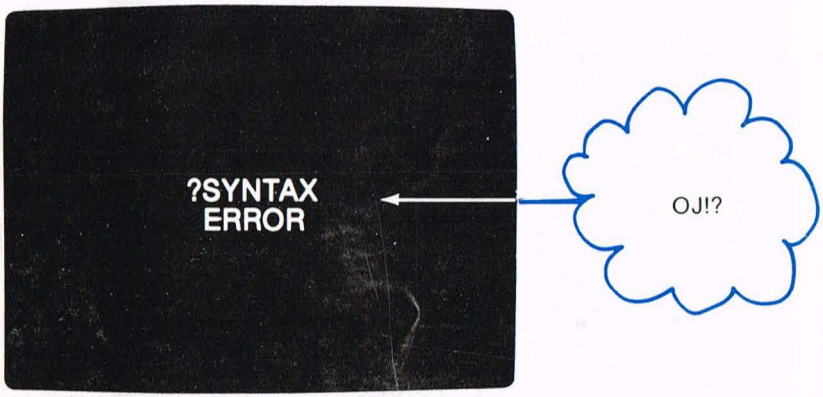
När du tryckte på RETURN tangenten talade du om för VIC att du skrivit färdigt. VIC börjar då se på vad du skrivit och utför ditt kommando. I detta kommando har du sagt att VIC skall skriva en text (print = skriva). VIC skriver då ut allting som finns mellan citationstecknen (RAINBOW).

När VIC skrivit ut ordet READY och markören blinkar, visar VIC att den är färdig att ta emot nya uppgifter.

Här är några fler texter som du kan få din VIC att skriva ut. Försök med dessa eller gör upp egna texter.



Experimentera med att använda andra tecken än bokstäver mellan citationstecknen. OBS Stavas PRINT fel så skrivs meddelandet SYNTAX ERROR ut. Det betyder att VIC inte förstår vad du skrivit.



Men detta är inget att oroa sig för, ingenting som du skriver in på tangentbordet kan skada VIC. Har du gjort ett skrivfel så säger VIC ifrån, genom att skriva ut ett meddelande. Dessa meddelanden och vad de betyder, finns beskrivna i Bilaga N. Om du får ett SYNTAX ERROR behöver du inte bry dig om det, utan fortsätt att skriva.

När du har skrivit en stund blir skärmen full av text. VIC har då en bra tangent för att rensa upp skärmen. För att rensa skärmen gör följande:

Håll nere SHIFT tangenten och tryck på CLR HOME tangenten.

VIKTIGT

Skärmen töms på all text som du och VIC skrivit och markören flyttar sig upp till översta raden längst till vänster.

Kom ihåg denna tangentfunktion, för den kommer du att använda ofta i fortsättningen när du blivit mera bekant med VIC.

DITT FÖRSTA PROGRAM

Om din VIC klarade av att skriva ut dina texter, hur knasiga dom än verkade vara, klarar den säkert av att du skriver in ditt första program.

STEG 1: Rensa skärmen genom att trycka SHIFT och CLR HOME tangenten.

STEG 2: Skriv N E W och tryck på RETURN tangenten.

STEG 3: Skriv 1 P R I N T " SPACE V I C 2 0 " ; och tryck RETURN

semikolon

Använd SHIFT tangenten för dessa

STEG 4: Skriv 2 G O T O 1 och tryck RETURN

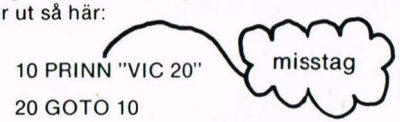
Skriv inte ut bokstäverna i detta ord — tryck på blanktangenten (den breda tangenten längst ner på VIC's tangentbord).

När du skrivit färdigt ser skärmen ut så här:

TIPS: Ändring av skrivfel i program

Om du gör ett skrivfel när du skriver ett program finns följande möjligheter att ändra:

1. Du kan när som helst skriva om en rad och VIC kommer omedelbart att byta ut dem gamla raden i programmet mot den nya raden. T ex om ditt program ser ut så här:



   Du korrigerar genom att skriva: RETURN

   10 PRINT "VIC 20" RETURN

   VIC har nu bytt ut den gamla mot den nya och för att försäkra dig om att så är fallet, skriv [L] [I] [S] [T], och programmet listas ut med ändringen utförd.

2. Om du vill radera en rad gör du detta genom att skriva radens nummer följt av RETURN. Ex: 10

3. Du kan ändra på en rad genom att flytta markören till det felaktiga tecknet och rätta det, och sedan trycka RETURN. Notera att citationstecken kan orsaka att du på skärmen får upp en del konstiga tecken. Tryck då RETURN, och försök igen.

4. Trycks SHIFT och INST DEL samtidigt kan du klämma in blanktecken till höger om markören, på en rad som redan skrivits.

5. Enbart INST DEL tangenten raderar tecknet direkt till vänster om markören.

Om allt på raden ser ut som det ska, tryck RETURN

Skriv:

[R] [U] [N] och tryck RETURN

Skärmen kommer att fyllas med VIC, och textraderna flyttas uppåt i skärmen.

Vill du sakta ner rullningen på skärmen, håll ner tangenten som sitter på tangentbordets vänstra sida och är märkt: CTRL

Med denna fantastiska tangent reglerar du hastigheten på utskriften.

Hur stoppar vi programmet då? Bra fråga! Leta rätt på följande tangent på tangentbordet: RUN STOP

Tryck på RUN STOP tangenten. Programmet stannar och följande meddelande skrivs ut:

BREAK IN 1

READY

Meddelandet betyder att programmet stannade på rad 1. VIC är nu åter redo att ta emot nya instruktioner från dig.

Låt oss nu se om ditt program fortfarande finns kvar. Skriv:

L I S T

Dina programrader kommer utskrivna på skärmen precis som du skrev in dem. Skriv nu RUN och programmet kör igång igen.

Du har nu använt de vanligaste kommandona för arbetet med VIC. Du har:

- PRINT skrivit ut text på skärmen.
- CLR rensat skärmen.
- CTRL fått långsammare utskrift.
- STOP stannat programmet.
- LIST fått en utskrift på programmet.
- Skrivit ditt första program och fått en rullande skärm.
- Lärt dig några sätt att redigera program.

Allt eftersom du går igenom resterande kapitel i denna bok kommer du att upptäcka att mycket av det som finns i detta kapitel återkommer. Har du några frågetecken efter detta kapitel så bry dig inte om det, utan fortsätt att gå igenom boken. Frågetecknen får säkert sin förklaring när du arbetat med VIC ett tag.

Denna bok är gjord så att du kan gå in i det kapitel som verkar intressant att läsa. Du behöver inte läsa varje kapitel för att lära känna din VIC. Var bara säker på att du läser från början av varje kapitel. Vi hoppas att denna bok skall hjälpa dig att få en spännande och lärorik färd genom datorernas värld.

2

Bildskärm och tangentbord

- Ditt första grafiska tecken
- En beskrivning av tangentbordet
- Att skriva på skärmen
- VIC som miniräknare
- Introduktion till färg

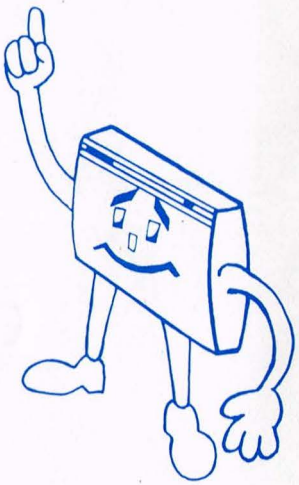
Skriv detta program

Skriv programmet precis som rutan nedanför visar, och se vad som händer!

```
10 PRINT " SHIFT CLR HOME
20 FOR T=1 TO 300: NEXT
30 PRINT "DITT NAMN"
40 FOR T=1 TO 300: NEXT
50 GOTO 10
```

Skriv [R] [U] [N] och tryck **RETURN**

För att stoppa programmet, tryck **RUN STOP** tangenten



Bildskärm och tangentbord

Detta kapitel förutsätter att du läst igenom och förstått kapitel 1: Lär känna din VIC 20. Har du inte gjort det gå tillbaks och läs åtminstone de två sista delarna i kapitlet, där tangentbordet och skärmhantering behandlas.

Håll ner SHIFT tangenten och tryck på CLR HOME tangenten.

N E W och tryck RETURN

1 P R I N T " H E J S A N SPACE

SPACE " ;

och tryck RETURN

2 G O T O 1

och tryck RETURN

R U N och tryck RETURN

Detta betyder att du skall trycka på den breda blanktangenten

Skärmen fylls med ordet: HEJSAN

Orden ser ut att röra sig uppåt och sidledes. Tryck på CTRL tangenten för att dra ner på hastigheten. När utskrift sker så ser du att innehållet i skärmen flyttas uppåt, för att skapa plats för mera text. Rörelsen uppåt är verklig. Sidorörelsen är däremot en illusion, som beror på antalet tecken som skrivs ut.

Stanna programmet genom att trycka på RUN STOP tangenten.

Skriv nu dessa två rader:

1 SPACE P R I N T " H E J S A N DITT NAMN " ;

och tryck RETURN .

R U N och tryck RETURN .

Ditt namn syns nu på TV-skärmen. Illusionen av rörelser beror på längden av ditt namn.

Vill du stanna programmet tryck på RUN STOP tangenten.

DITT FÖRSTA GRAFISKA TECKEN

Rensa skärmen (håll ner SHIFT och CLR HOME). Håll nere SHIFT och tryck S . Du skall nu få ett blått hjärta på skärmen. Du har nu skrivit ditt första grafiska tecken.

Vi skriver några andra grafiska tecken. Håll nere C= tangenten och tryck samtidigt på några andra tangenter — detta blir det vänstra grafiska tecknet på tangenten. Grafiska tecken är bra när tabeller och kurvor skall ritas ut på skärmen. Trycker du SHIFT och C= samtidigt, kan både stora och små bokstäver skrivas. Se kapitel 3 för en förklaring.

Skärmens storlek

Hur många tecken får plats i skärmen på en VIC? Låt oss undersöka saken. Rensa skärmen och skriv följande:

N E W och tryck RETURN

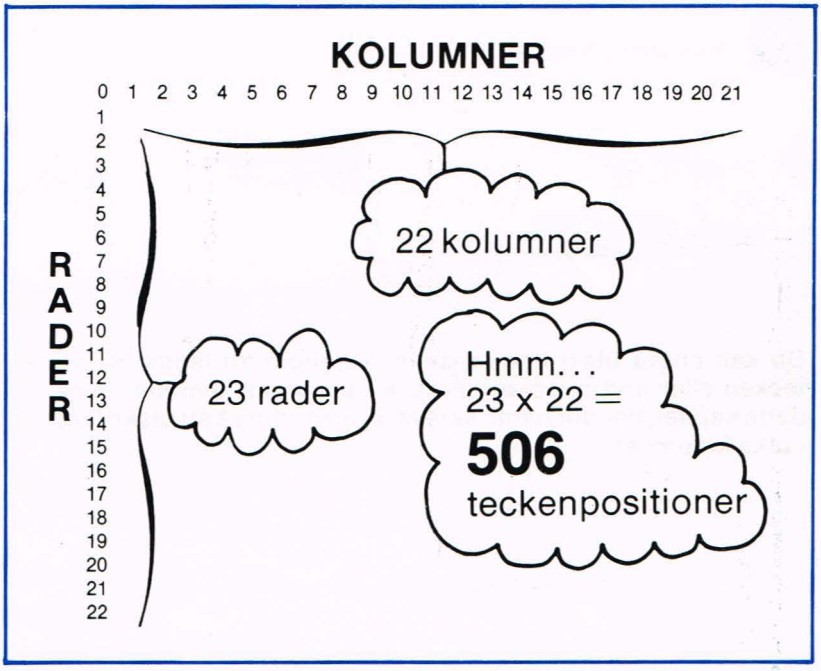
1 P R I N T " ♥ " ; RETURN

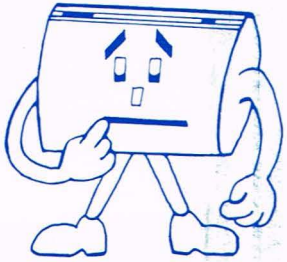
För att skriva ett blått hjärta, håll nere SHIFT och tryck S

2 G O T O 1 RETURN R U N RETURN

Skärmen fylls med blå hjärtan. Räkna hur många hjärtan det får plats på en rad. Det finns 22 stycken. VIC har 22 teckenpositioner på bredden (horisontellt). Positionerna längs raden kallas ibland för kolumner. VIC har 22 kolumner.

Hur många rader finns då på skärmen. Tryck på CTRL tangenten så att utskriften går långsammare. När CTRL hålls nedtryckt fladdrar de fyra sista raderna på skärmen. VIC har 23 rader på skärmen (lodrätt).



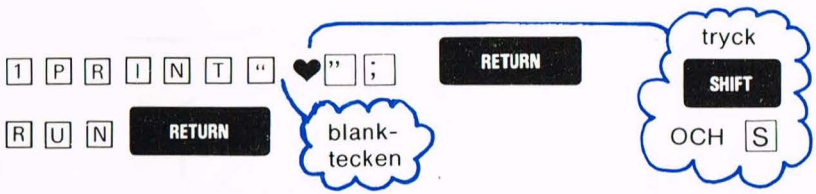


VIC har plats för 506 tecken på skärmen, vilka kan vara bokstäver, siffror eller grafiska tecken. Man kan säga att VIC kan trolla med 506 tecken på samma gång. Fantastiskt, eller hur?

TIPS

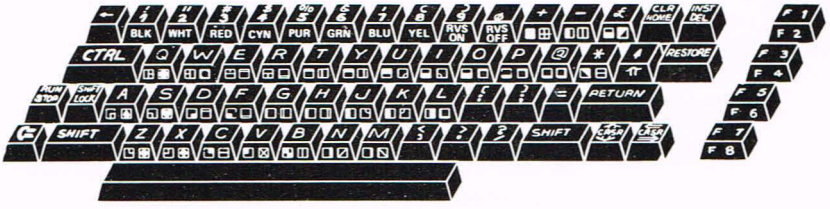
Eftersom VIC har 22 kolumner i skärmen, kommer texter som är jämnt delbara med 22 (2, 11, 22 tecken långa) att skrivas ut i jämna och fina kolumner. Texter som inte har dessa längder kommer att "rinna" över till nästa rad. Pröva själv så får du se.

För att stoppa VIC att skriva blå hjärtan i skärmen tryck på RUN STOP tangenten. Skriv sedan:



Du kan ändra utskriftens utseende genom att lägga till blanktecken eller andra tecken. Försök i programexemplet tidigare i detta kapitel, där ditt namn skrevs ut, att ändra så att utskriften blir i raka kolumner.

EN VANDRING PÅ TANGENTBORDET



Du har hittills använt tangentbordet för att skriva texter, grafiska tecken, kontrollerat VIC (CTRL och RUN STOP), och ändrat i texter (INST DEL).

STARTTANGENT

Trycker du RUN STOP och RESTORE samtidigt, nollställer du VIC precis som när du slår på den...med undantag att programmet som fanns i maskinen inte förstörs vid starten.

SHIFT

SHIFT tangenten — VIC's tangentbord ser ut precis som på en skrivmaskin och har en tangent för att låsa SHIFT (SHIFT LOCK). SHIFT används tillsammans med andra tangenter för att få en annan funktion eller skriva ett annat tecken.

CLR HOME

CLR HOME tangenten — Trycker du på denna tangent så flyttar sig markören till skärmens övre vänstra hörn (home positionen). Om SHIFT tangenten varit nedtryckt samtidigt hade skärmen dessutom rensats från tecken.

CRSR CRSR

CRSR tangenter — Markör-tangenten (CRSR = cursor = markör) används för att förflytta markören uppåt, nedåt eller sidledes. Markör-tangenterna har en repeterande funktion, dvs hålls tangenten nedtryckt upprepas funktionen och markören fortsätter att vandra på skärmen. Varje tangent har en uppsättning pilar som visar åt vilket håll markören vandrar. Vill du flytta markören nedåt eller till höger trycker du respektive tangent. Vill du flytta den uppåt eller vänster håller du nere shift och trycker på respektive tangent. När markören flyttar sig över tecknen på skärmen förstörs inte något tecken.

RETURN

RETURN tangenten — Du trycker RETURN när du skrivit färdigt varje programrad. RETURN betyder för VIC att den ska läsa in en rad eller utföra ett kommando. För att förstå funktionen kan det kanske hjälpa att tänka sig RETURN-tangenten som en inmatningstangent, den stoppar information eller kommandon till datorn.

CTRL

CTRL tangenten — Denna tangent används tillsammans med färgtangenterna för att ändra färgen på tecknen. Tangenten kan också användas för att definera egna tangentfunktioner. En del av plugg-in modulerna använder CTRL-tangenten för att få specialfunktioner. CTRL-tangenten fungerar som SHIFT-tangenten. Den hålls nedtryckt medan du trycker på en annan tangent.

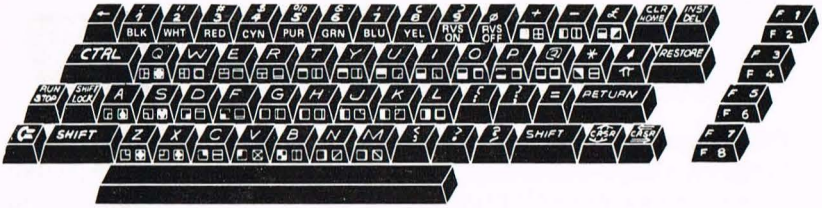
Färgtangenter — Färgen på tecknen kan ändras genom att CTRL-tangenten hålls nedtryckt och en av de 8 färg-tangenterna på den övre raden av tangentbordet trycks ned. Det finns en förkortning av varje färg på tangenterna. Färgerna är svart, vit, röd, cyan (ljusblå), violett, grön, blå, gul. Med dessa tangenter kan färgen bestämmas eller ändras på tecken som skrivs av dig eller VIC. Alla tecken du skriver blir i den färg du senast bestämde.

RVS ON och RVS OFF tangenterna — Du kan få tecknen omvända (reverserade) på skärmen om du trycker CTRL och RVS ON. Allt som du skriver kommer att bli reverserat, dvs tecknet får bakgrundens färg och bakgrunden får tecknets färg. För att skriva normala tecken tryck CTRL och RVS OFF.

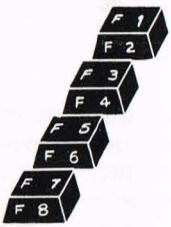
RUN STOP tangenten — Trycker du på denna tangent slutar VIC med sitt arbete och för över kontrollen tillbaks till dig. Kör VIC ett program, kan du stanna programmet med denna tangent. Håller du SHIFT nere och trycker på tangenten, laddar VIC in information i minnet från en yttre kassettstation.

INST DEL tangenten — Du kan lägga in blanktecken eller radera tecken med denna tangent. Enbart tryckning på tangenten raderar tecknet direkt till vänster om markören. Tecknen till höger om markören flyttas ett steg till vänster. Används SHIFT till tangenten läggs blanktecken in till höger om markören. Tecken till höger om markören flyttas ett steg till höger. Befinner sig markören mitt i en textrad kan du med SHIFT och tryckning på tangenten klämma in blanktecken i textraden. Denna funktion är mycket användbar vid redigering av program.

Grafik och Commodore tangent — När VIC sätts på är den automatiskt i grafikläge, vilket betyder att när du trycker SHIFT skrivs det grafiska tecken på skärmen. Det finns ca 60 grafiska tecken. På varje tangent finns två grafiska tecken. Vill du skriva det högra grafiska tecknet tryck då samtidigt SHIFT och för att skriva det vänstra hålls Commodoretangenten nedtryckt. På detta sätt kan du skriva stora bokstäver och alla grafiska tecken på samma gång. Med grafiken kan du skapa bilder, figurer, spelkort osv.



Små bokstäver eller grafik — Trycker nu ned SHIFT och COMMODORE-tangenterna samtidigt kommer du i textläge. I detta läge kan du skriva både stora och små bokstäver plus grafiken på tangenternas vänstra sida. För att komma tillbaks till ursprungsläget tryck åter på SHIFT och COMMODORE-tangenterna samtidigt.



Programmerbara funktionstangenter — De fyra tangenterna på tangentbordets högra sida är inte definierade när VIC startar upp. Tangenterna kan användas till att utföra olika funktioner i program. Tillsammans med SHIFT kan du få 8 stycken olika funktioner. Tangenterna används mest i plugg-in korten med specialprogram, men även en programmerare kan använda dessa tangenter.

Specialtangenter — VIC tangentbord innehåller dessutom tangenter som inte finns på skrivmaskiner. Till och med en del datorer saknar dessa. Ex pi (π), pil vänster (←), pil upp (↑), större/mindre än ($>$ $<$).

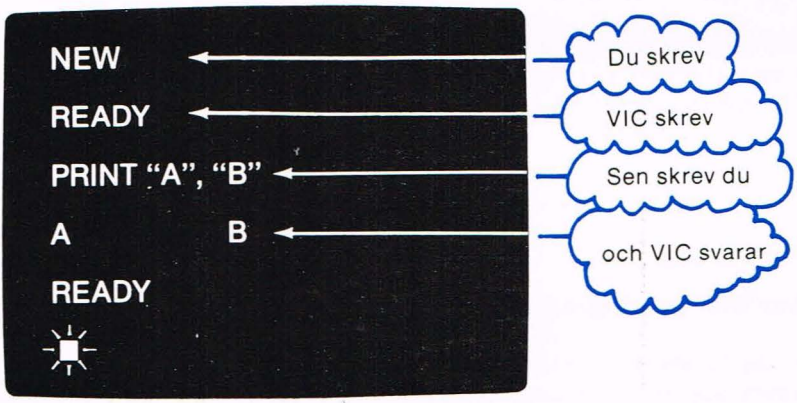
Detta avslutar din vandring på VIC's tangentbord. Det är väldigt svårt att med ord förklara alla finesser med VIC's tangentbord. Det bästa sättet att lära sig är att på egen hand experimentera och leka sig fram. Pröva de olika textlägena, lek med den rikhaltiga grafiken. Kom ihåg att tangentbordet är länken mellan datorn och dig. Behärskar du tangentbordet så blir all kommunikation med VIC mycket lättare.

Rensa skärmen och skriv in följande rader:

NEW **RETURN**

PRINT "A", "B" **RETURN**

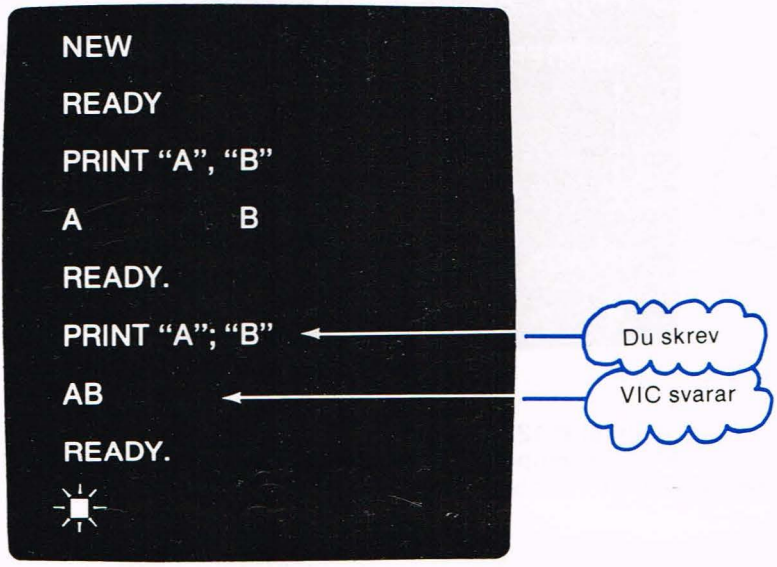
Skärmen visar:



Skriv nu in denna rad:

RETURN PRINT "A", "B"

Skärmen visar



När ett kommandotecken används för att separera A och B i första raden, skrev VIC ut bokstäverna med ett antal blanktecken mellan dem. Används istället semikolon skrivs bokstäverna ut utan extra blanktecken.

I den första utskriften var det 11 blanksteg mellan de båda tecknen. Detta ger oss en ledtråd till anledningen. VIC delar upp skärmen i två lika stora delar.

När VIC skriver ut två texter eller numeriska tal separerade med kommatecken, skrivs det första objektet ut på skärmens vänstra del och det andra på den högra...

den första är längre än 11 tecken.

Är det första objektet kortare, eller lika med 11 tecken långt, skrivs nästa objekt ut på samma rad med början mitt på skärmen. Är första objektet längre än 11 tecken, skrivs det andra objektet ut på nästa rad. Rensa skärmen och försök med detta exempel:

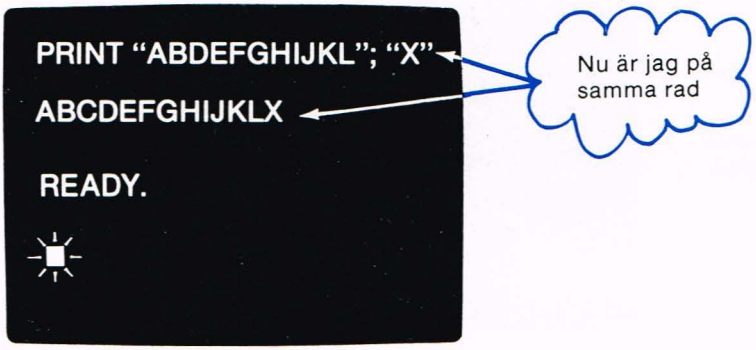
```
PRINT "ABCDEFGHIJKL", "X"
```

Skärmen visar:

Det första objektet är 12 tecken långt, så "X" skrivs ut på nästa rad. Upprepa samma exempel men med semikolon istället för kommatecken mellan de två objekten.

PRINT "ABCDEFGHIJKL"; "X"

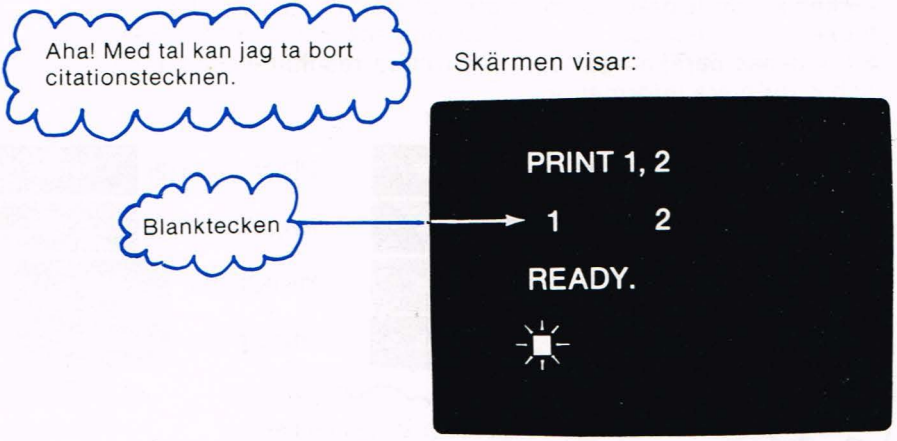
Skärmen bör visa:



Förstår du principen? VIC uppför sig som en skrivmaskin med en automatisk tabulator satt i skärmens mitt. När VIC ser ett kommatecken hoppar den fram till skärmens mitt eller till början på nästa rad, det som blir närmast.

Rensa skärmen och skriv följande:

PRINT 1, 2

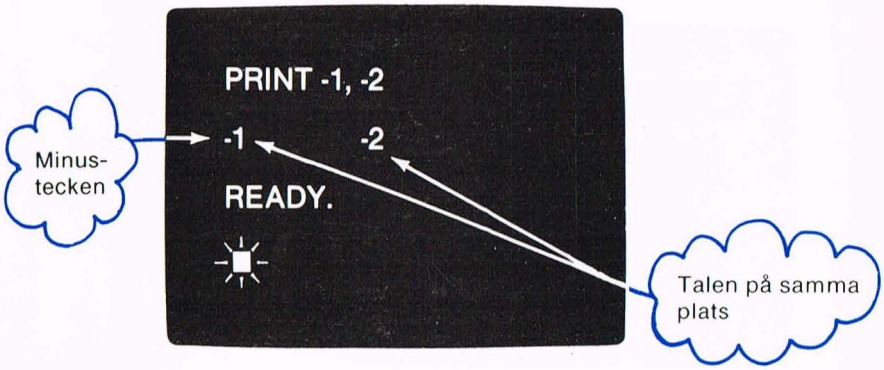


Ser du blanktecknet före första siffran? När VIC skriver ut tal lämnar den ett blanktecken i början för att ge plats för eventuellt minustecken. Är talet positivt skrivs ett blanktecken ut istället. Negativa tal inleds med ett minustecken.

Försök och se. Ändra raden till:

PRINT -1, -2

Se på skärmen vad som händer:

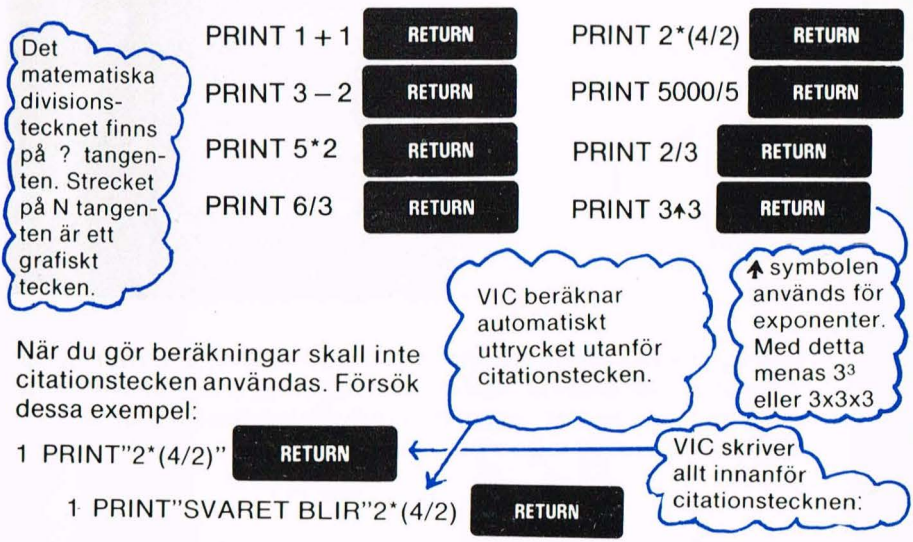


Siffrorna skrivs ut på samma ställen men de inleds med minustecken.

Dessa exempel hoppas vi har givit dig en inblick i hur texter och information kan presenteras på skärmen. Det finns många olika sätt att göra detta. Fler kommer du att lära dig när du fortsätter med din VIC.

VIC som miniräknare

VIC kan också användas som en 9-siffrig miniräknare. + och — tecken används precis som i matematiken. VIC's multiplikationstecken är en asterisk (*) och divisionstecken ett snedstreck (/). Skriv dessa beräkningar och kontrollera resultatet. Se bilaga C och K för mera information.

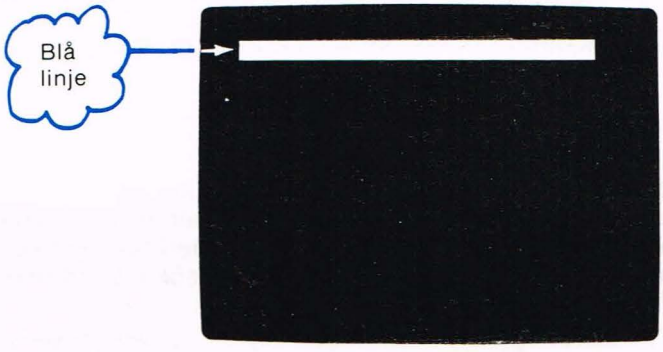


EN INTRODUKTION TILL FÄRG

VIC kan skriva alla tecken i 8 olika färger. Tecknen kan dessutom reverseras.

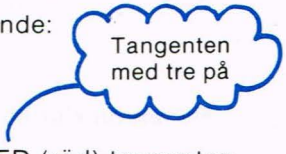
Rensa skärmen, håll ner CTRL och tryck på denna tangent:

Släpp alla tangenter och tryck ned den långa blankstegstagneten och håll den nedtryckt. Vad händer? Blir det en tjock blå linje över skärmen?



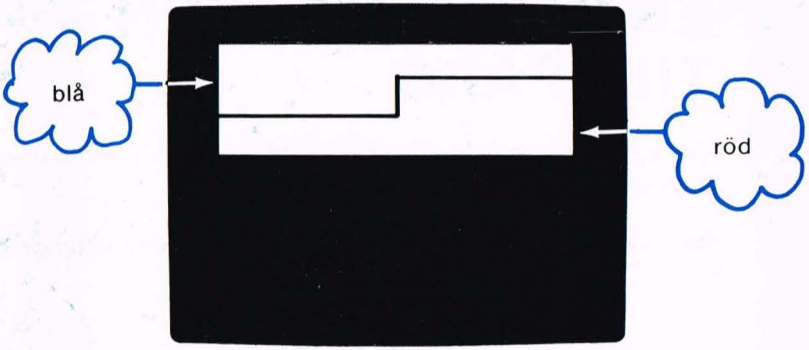
Håll nere tangenten så länge du vill. När markören kommit till radens slut hoppar den ned till nästa rad och fortsätter där med att bilda en ännu bredare linje.

Släpp blankstegstangenten och gör följande:



Håll ner CTRL tangenten och tryck på RED (röd) tangenten.

Markören blir nu röd. Tryck ned blankstegstangenten och fortsätt. Nu ändrar linjen färg till rött. Va skoj! Fortsätt att måla!



Ändra på färgerna som du vill. Gör olika långa och breda linjer. Upptäck VIC's fina färgmöjligheter som sprider litet färgglädje i ditt liv.

Skriv nu CTRL RVS OFF och tryck på blankstegstangenten.

Ingenting händer förutom att markören flyttar på sig.

Skriv CTRL RVS ON

och dina linjer kommer tillbaka när du trycker på tangenten. Försök att skriva några bokstäver. Reverserad text är bra till rubriker och för att märka ut speciell text. Du kan även använda reverserade tecken i program. Försök följande exempel:

```
NEW

10 PRINT " CTRL RVS ON VIC20";
20 GOTO 10
RUN
```

håll denna nere medan du trycker

glöm inte att trycka RETURN efter varje rad.

För att bli klar till nästa kapitel, skriv:

RUN STOP RESTORE och ordet NEW RETURN

Fr o m nu så använder du denna metod för att radera oönskade program och börja om på nytt.

Färg och grafik

- Programmering av färg
- Färgtangenter
- Bakgrunds- och ramfärger
- Kombination av bakgrunds- och ramfärger
- Färg till tecken
- Teckenpositioner
- Slumpfärger
- Ljud och färg
- Grafik på tangentbordet
- Grafik i överskrifter och rubriker

Skriv detta program:

Skriv programmet precis som rutan nedanför visar, och se vad som händer!

```
1FOR H = 1 TO 505
2PRINT " ♥ ";
3NEXT
4FORC = 8 TO 255 STEP 17
5POKE 36879, C
6FOR T = 1 TO 500: NEXT
7NEXT
8GOTO 50
RUN
```

Detta betyder att 505 hjärtan skrivs ut på skärmen.

Detta ändrar värdet på C (color = färg) 17 steg i taget.

Detta är tidsfördröjningsloop som talar om för VIC att räkna till 500 innan den byter färg igen.

Skriv RUN och tryck

För att stoppa programmet, tryck RUN STOP tangenten.

FÄRG OCH GRAFIK

En VIC och en färg-TV ger dig möjlighet att lägga in färger var som helst på din TV-skärm. När du startar upp din VIC har ramen, markören och alla tecken redan fått färg. VIC kan visa 8 markörfärger (teckenfärger), 8 ramfärger och 16 bakgrundsfärger.

Låt oss börja med att använda VIC's tangentbord för att göra färger på skärmen. Skriv en bokstav. Bokstaven visas med mörkblå färg och vit bakgrund på skärmen. Titta nu på den översta raden på tangentbordet (tangenterna 1 till 8). På framsidan på tangenterna finns en engelsk förkortning av de olika färgerna. Leta rätt på tangenten märkt CTRL på den vänstra sidan.

Håll CTRL tangenten nedtryckt och tryck tangenten märkt YEL. (gul).

Släpp CTRL och YEL tangenterna och skriv en bokstav. Denna bokstav får nu gul färg. Håll ner CTRL tangenten och tryck på en annan färgtangent. Skriv sedan några bokstäver. Du ser hur lätt det är att ändra färg på text.

Leta rätt på tangenten märkt RVS ON. Håll ner CTRL tangenten och tryck på RVS ON. Skriv några bokstäver. Alla bokstäver du skriver (tills du trycker på RETURN tangenten) visas i reverserad form på skärmen, som ett fotografiskt negativ. Håller du ner CTRL tangenten och trycker på RVS OFF, visas bokstäverna normalt när du skriver dem.

PROGRAMMERA I FÄRG

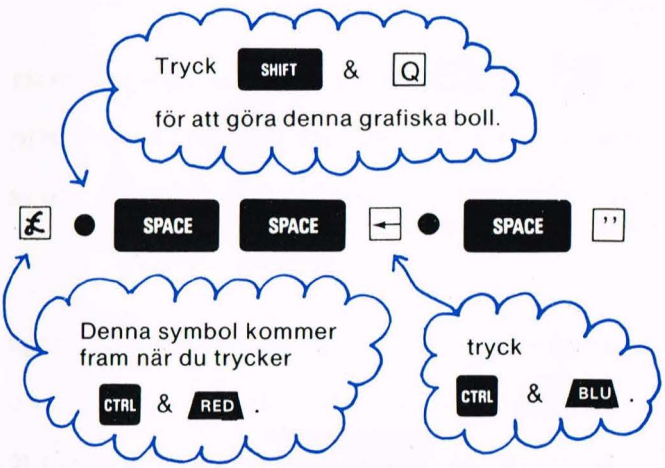
Låt oss kombinera färgkontrollen med några enkla program-kommandon. Notera att när du trycker CTRL och en färgtangent mellan citationstecknen visas ett reverserat grafiskt tecken istället! Detta är helt OK.

Håll ner SHIFT tangenten och tryck CLR HOME.

Skriv:

N E W och tryck RETURN

1 P R I N T " SPACE



och tryck RETURN

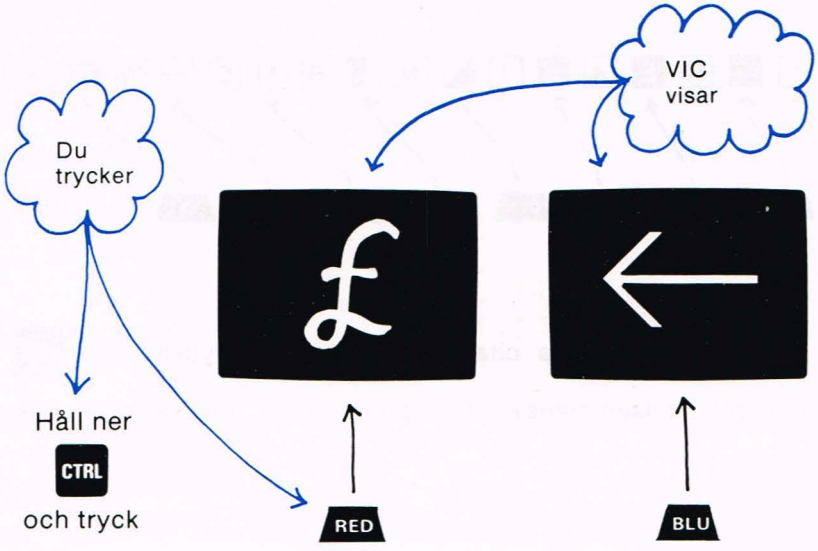
Skriv: 2 SPACE G O T O SPACE 1

och tryck RETURN

Har du problem med dessa exempel, hoppa då tillbaka till första kapitlet "Lär känna din VIC". Du kan inte skada din VIC med något du skriver, men du kanske blir konfunderad över vissa tangentfunktioner. Om du av misstag låter SHIFT LOCK vara nedtryckt, blir texten du försöker skriva en rad med grafiska tecken. Gör du ett misstag, tryck RETURN några gånger och skriv om hela raden. Den nya raden ersätter automatiskt den gamla. När du har dessa två rader på skärmen, skriv:

R U N och tryck RETURN .

När du tryckt det sista RETURN , kommer du att se hundratals röda och blå bollar flyta omkring på skärmen. Se tillbaks på exemplet. Du ser de två konstiga tecknen i rad 1. De skapades när du tryckte CTRL tillsammans med RED och BLU tangenterna. Dessa symboler är VIC's förkortning för färgförändring.



Inget märkvärdigt. När du tröttnat på rött och blått, tryck STOP. Om du vill kan du experimentera fram andra färger genom att skriva om rad 1.

Du kan med litet fantasi skapa ett fascinerande färgspel.

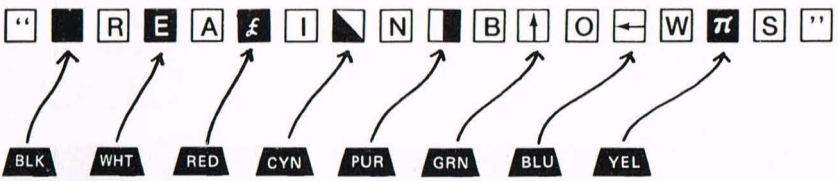
VIC:s FÄRGTANGENTER

I det sista exemplet upptäckte du att du kunde lägga in färgkontroll i dina utskrifter genom att hålla CTRL tangenten nedtryckt och trycka på en av dessa tangenter:

BLK WHT RED CYN PUR GRN BLU YEL

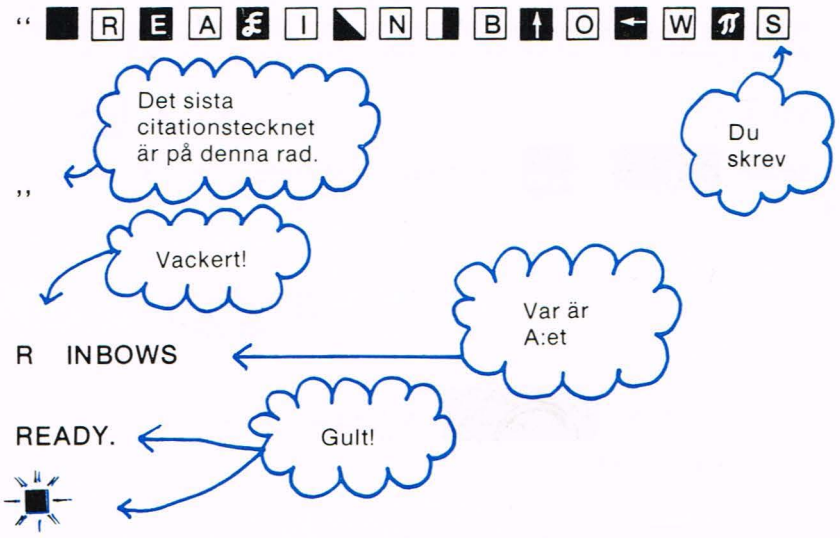
Tangenterna är numrerade från 1 till 8. När du trycker på dessa tangenter mellan citationstecknen i en PRINT-sats, dyker det upp konstiga tecken på skärmen. För att se hur alla dessa tecken ser ut stannar du programmet, rensar skärmen och skriver:

```
PRINT
```



Glöm inte att skriva citationstecknen och trycka ned CTRL tangenten tillsammans med färgtangenterna. Din skärm bör visa:

PRINT



Var är bokstaven "A"? Den andra färgtangenten är vit och bakgrunden är... rätt gissat... VIT. Ett vitt "A" på en vit bakgrund ger oss ett blanktecken i RAINBOWS.

De andra sju bokstäverna i ordet RAINBOWS är alla av olika färg. Den sista bokstaven S, är gul. Notera att även meddelandet READY och markören är gula. VIC skriver tecken i den sist angivna färgen.

För att ändra färgen på markören till den vanliga färgen tryck CTRL och BLU. Om du vill kan du försöka skriva egna texter i olika färger. Vi kommer att behandla färgtangenterna mera ingående strax, men först ett viktigt meddelande...

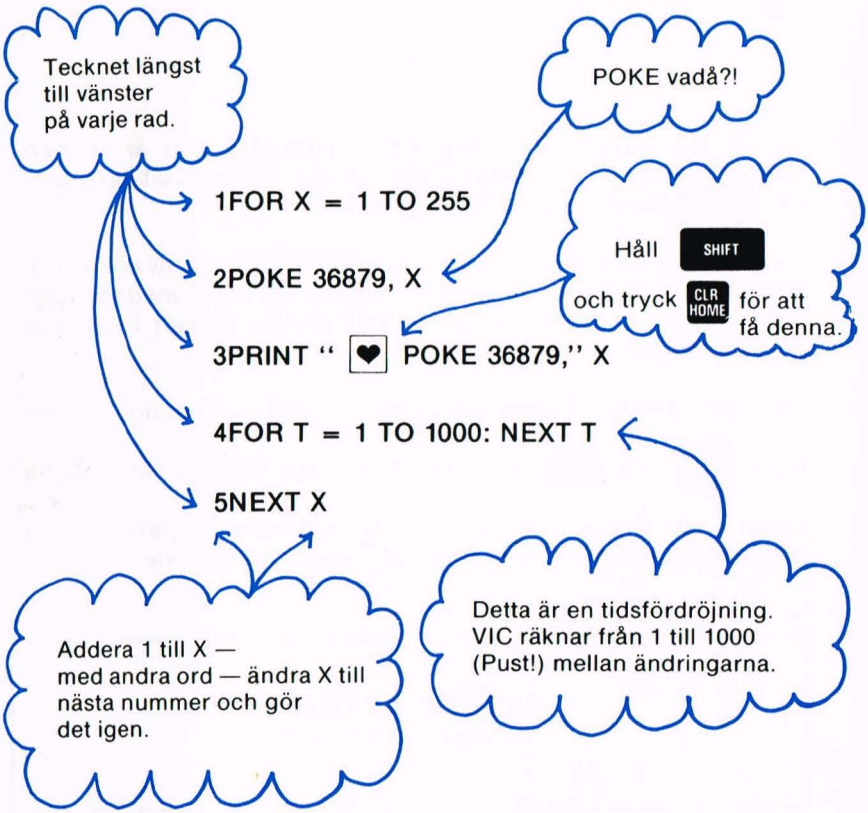
EXTRA!! EXTRA!!
VIC's färgshow är på väg!!

BAKGRUNDS- OCH RAMFÄRGER

Nu när du vet hur färgerna på tecknen ändras, ska vi visa hur färgerna på bakgrunden och ramen kan ändras. Du kan få 255 olika kombinationer av bakgrunds- och ramfärger. För att försäkra dig om att VIC är klar för våra exempel, tryck RUN STOP och sedan SHIFT , CLR HOME för att rensa skärmen. Skriv sedan in dessa rader i VIC:

N E W och tryck RETURN .

(Tryck RETURN när du skrivit färdigt varje rad.)



När du skrivit dessa rader, kontrollera att de stämmer överens med exemplet. Om de inte stämmer överens, skriv då om raden från början. Vill du inte skriva om hela raden kan du med hjälp av

CRSR tangenten flytta upp markören till den felaktiga raden och

skriva om den. Kom ihåg att trycka RETURN när du ändrat färdigt raden.

När du är nöjd med dina rader tar du fram papper och penna. Skriv sedan:

R U N och tryck RETURN

Din skärm kommer att spraka med olika färgkombinationer. Bakgrunden ändrar färg. Ramen ändrar färg. Tecknen på skärmen ändrar också färg. VIC presenterar 255 stycken olika färgkombinationer. (Får du SYNTAX ERROR i någon rad, skriv om den och skriv sedan:

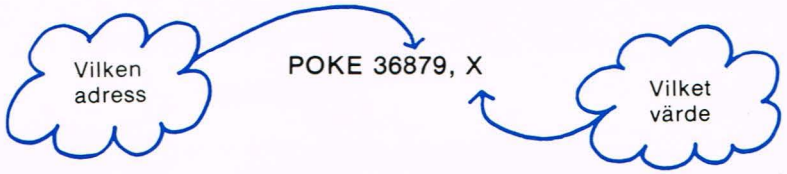
R U N och tryck RETURN igen.)

Om du upptäcker någon färgkombination som verkar intressant skriv ner numret på den kombinationen. Endast det sista numret i utskriften ändras. Texten POKE 36879, förblir densamma. (Ibland går inte texten att läsa. Det beror på att tecknen får samma färg som bakgrunden. Texten blir synlig igen efter några färgbyten.) VIC har 8 ramfärger, 16 bakgrunden och 8 teckenfärger. Alla 8 teckenfärgerna kan användas på vilken bakgrund som helst. Detta ger möjligheter till många vackra kombinationer.

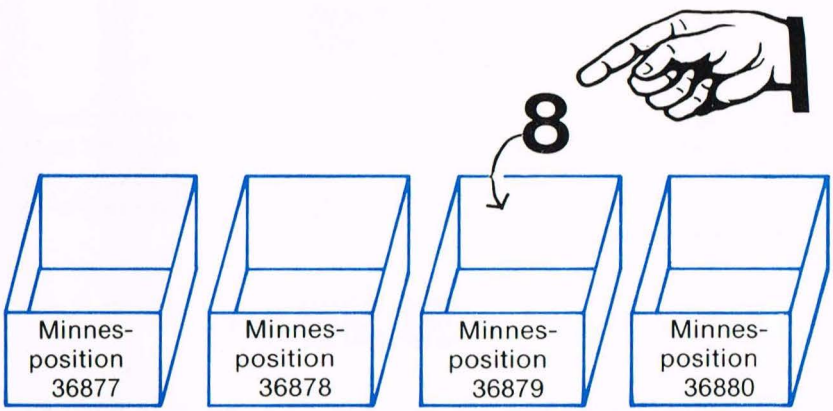
Vill du ha tillbaks skärmen i sina ursprungliga färger tryck ner

RUN STOP tangenten och håll den nedtryckt. Tryck samtidigt på RESTORE

Rad 2 i exemplet ovan gör att skärmen ändrar färg. Raden innehåller ett POKE kommando. Varje POKE kommando har två stycken värden som det använder:



Det första numret (i detta fall 36879) är adressen till minnespositionen (tänk dig en liten box märkt med 36879) där det andra numret (X) skall läggas in. Positionen 36879 råkar vara den position där VIC lagrar information om vilka färger som skärmens olika delar skall ha. Varje värde på X motsvarar en färgkombination. I detta exempel börjar X med värdet 1 och ändras sedan till 2, 3 osv upp till slutvärdet 255.



ETT EXEMPEL PÅ POKE 36879,8

För att hjälpa dig att hitta den perfekta färgkombinatinen har vi sammanställt en tabell där de olika bakgrundsfärgerna och ramfärgerna ger ett "X" värde. Tabellen ger värdet som skall användas tillsammans med POKE 36879 för att få den önskade färgkombinationen. De värden som saknas gör att alla tecken i skärmen reverseras.

BAKGRUNDS- OCH RAM-FÄRGER

	Ram							
	Svart	Vit	Röd	Cyan	Viol.	Grön	Blå	Gul
Bakgrund								
Svart	8	9	10	11	12	13	14	15
Vit	24	25	26	27	28	29	30	31
Röd	40	41	42	43	44	45	46	47
Cyan	56	57	58	59	60	61	62	63
Violett	72	73	74	75	76	77	78	79
Grön	88	89	90	91	92	93	94	95
Blå	104	105	106	107	108	109	110	111
Gul	120	121	122	123	124	125	126	127
Orange	136	137	138	139	140	141	142	143
Lj. Orange	152	153	154	155	156	157	158	159
Skär	168	169	170	171	172	173	174	175
Lj. Cyan	184	185	186	187	188	189	190	191
Lj. Violett	200	201	202	203	204	205	206	207
Lj. Grön	216	217	218	219	220	221	222	223
Lj. Blå	232	233	234	235	236	237	238	239
Lj. Gul	248	249	250	251	252	253	254	255

FÄRG PÅ SKÄRMEN

Det går att kombinera VIC:s färg och grafikmöjligheter till en sprakande färgskärm. Än en gång kommer du bara att skriva in några rader i VIC och upptäcka att den gör en väldig massa saker.

Har du händelsevis något exempel som fortfarande hoppar omkring i skärmen, tryck då RUN STOP. Rensa skärmen. Skriv sedan denna rad:

N E W

```
1 L = INT(RND(1)*500) + 1
```

Genererar ett slumptal mellan 1 och 500.

```
2 C = INT(RND(1)*8) + 1
```

Läs om dessa i kapitel 7.

Slumptal mellan 1 och 8

```
3 POKE 7680+L, 160
```

Var är denna position?

```
4 POKE 38400+L, C
```

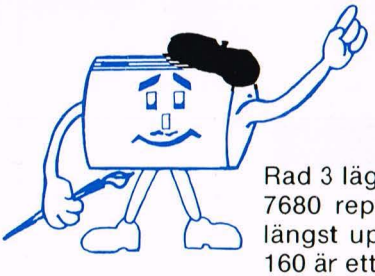
Och denna?

```
5 GOTO 1
```

Kontrollera att raderna blivit rätt inskrivna. Om de är korrekta, rensa skärmen och skriv [R] [U] [N] . Din skärm förvandlas till ett gytter av olikfärgade rektanglar.

VIC har gjort det igen! Med några få programrader kan du få detta fascinerande färgspel på din skärm. Hur fungerar då detta program?

Förenklat kan det sägas att rad 1 och 2 slumpvis väljer (generar) var på skärmen och vilken färg som skickas dit (P = Position och F = Färg). Rad 1 genererar ett nummer från 1 till 500. Rad 2 ger ett nummer från 1 till 8.



Rad 3 lägger in ett tecken på skärmen. Värdet 7680 representerar "HOME" positionen, dvs längst upp till vänster. Värdet som läggs dit, 160 är ett reverserat blanksteg.

Rad 4 lägger in färgen på tecknet. Värdet 38400 representerar motsvarande position i färgminnet, som 7680 gör i skärmminnet. Dvs "HOME" positionen.

SKÄRMPOSITONER

VIC TIPS

För att lägga in ett tecken i skärmen med hjälp av kommandot POKE så måste värden läggas in i både skärmminnet och färgminnet. Skärmminnet har sin första position i adressen 7680. Färgminnet har sin i 38400. Se bilaga för mer information.

Försök att lägga in andra tecken på skärmen. Ändra på värdet 160 i rad 3, till någonting mellan 0 och 255.

EXPERIMENTERA — njut av VIC:s färgsprakande personlighet.

Det finns 506 positioner på skärmen. Tänk dig varje position som en låda, och på varje låda finns ett nummer, lådans position. I var och en av dessa lådor kan du lägga in vilket tecken du vill och när som helst.

Nu har vi använt kommandot POKE tillräckligt. Tillbaks till färgtangenterna.

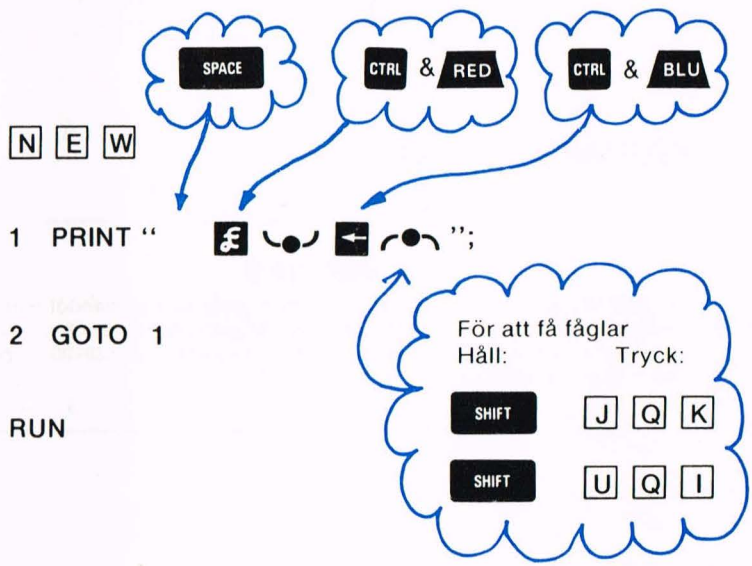
VIC TIPS

För att få tillbaks VIC's skärm till normala bakgrunds- och ramfärger skriv denna rad:

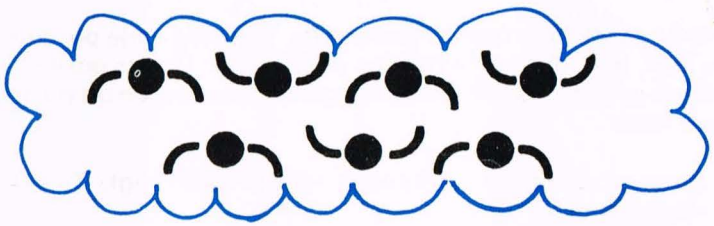
POKE 36879,27 RETURN

SLUMPVIS UTVALDA FÄRGER

Nu skall vi lära dig hur du kan få VIC att välja den färg den ska använda. Rensa först skärmen. Skriv sedan följande rader och tryck RETURN efter varje rad.



Din skärm fylldes med röda och blå fåglar. Undrar om de kan flyga också. Det ser ut som om de flyger.



Låt fåglarna flyga ett tag. Tryck RUN STOP när du inte vill att de skall flyga längre.

Nu vet du hur fåglar ritas på skärmen. Rensa skärmen och skriv in dessa rader (glöm inte att trycka RETURN):

N E W

1 A$ = " ■ E ƒ ◥ ◨ ↑ ← π "

För att skriva denna rad, håll CTRL nere och tryck på alla färgtangenter, en i taget. (BLK WHT RED CYN PUR GRN BLU YEL).

2 N = INT(RND(1)*8) + 1

Denna rad ger ett slumptal mellan 1 och 8

3 B$ = MID$(A$,N,1)

Färg-tecken

Slump-tal

Plockar ett färg-tecken ur A$.

4 PRINT B$ " ◡●◡ ◜●◝ ";

SPACE en gång.

5 GOTO 2

Kontrollera att du skrivit in raderna rätt. Är de korrekta? Rensa då skärmen och skriv [R] [U] [N] . Skrivs ett felmeddelande ut, skriv om programmet och försök igen med: [R] [U] [N] . Fåglar!

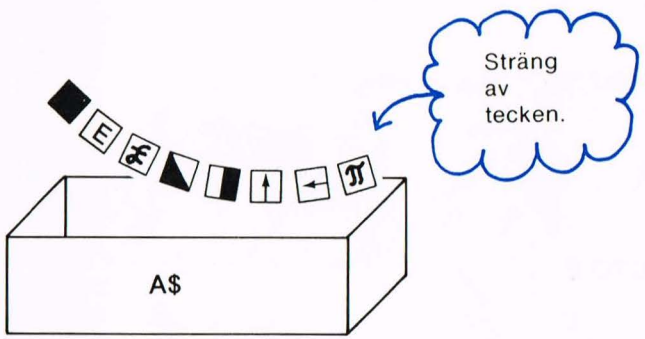
Fåglar! Överallt! I regnbågens alla färger. Notera att det finns luckor i din fågelflock. Luckorna är faktiskt vita fåglar. Vill du se dessa fåglar kan du ändra på bakgrundsfärgen. (Du kommer väl ihåg hur? Använd kommandot POKE i positionen 36879 tillsammans med ett lämpligt värde.)

Tänk på att om du färgar bakgrunden blå, försvinner alla blå fåglar. Försök så får du se. När du vill att fåglarna inte skall flyga längre tryck på RUN STOP .

Det sista exemplet använder en del av de inbyggda finnesser VIC innehåller för att få fram fåglar i olika färger på skärmen. Låt oss titta på exemplet rad för rad. Den första raden är:

1 A$ = " ■ E £ ◣ ◨ ↑ ← π "

Färgtangenterna formar en rad med grafiska tecken. Raden med grafiska tecken kallas för STRÄNG på datorspråk. Denna rad med tecken placeras i minnet med ett namn A$ (A-sträng). Du kan tänka dig A$ som en låda, där en sträng med tecken kan lagras. Det första tecknet i lådan A$ är den svarta färgtangenten. Det sista är den gula färgtangenten.



Nästa rad är:

```
2 N = INT(RND(1)*8) + 1
```

Denna rad ger ett slumpmässigt heltal mellan 1 och 8. Detta tal lagras i variabeln N. N är namnet på ett utrymme i VIC där tal kan lagras. VIC vet att N betyder att det är ett tal eftersom $-tecknet saknas.

I rad 1, används namnet A$ ($-tecknet alltid som sista bokstav) för att lagra färgtangenternas tecken. VIC vet att när den får namn som slutar med ett $-tecken så är det en sträng med tecken.

Vill du veta mera om [I] [N] [T] och [R] [N] [D] titta då i kapitel 7. Nu behöver du bara veta att raden ger ett tal mellan 1 och 8. Ser du 8:an till höger om likhetstecknet? Numret ger omfånget på talen. Ändrar du talet till 6, så ger raden ett tal mellan 1 och 6.

Hmmm... nästa rad:

```
3 B$ = MID$(A$,N,1)
```

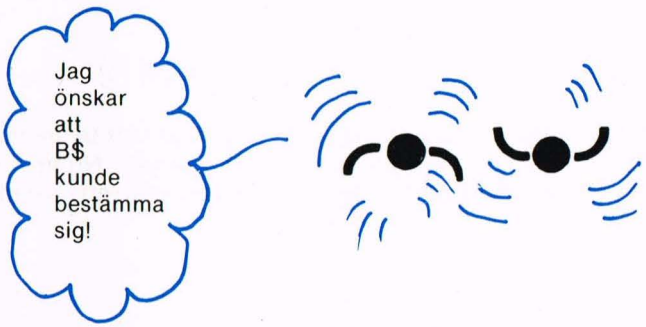
Denna rad ger plats i minnet för en låda kallad B$, och lägger ner ett färgtangenttecken i den. A$ innehåller alla färgtangenterna medan B$ endast innehåller ett färgtangenttecken. Vad gör då MID$? Den plockar ut ett tecken från A$ (det n:te tecknet) och lägger in det i B$. Nu har vi alltså fått ihop till tre stycken lådor. En kallar vi för A$ och en annan för N och den tredje för B$. Varje gång VIC kommer till rad 2 får den ett nytt värde på N. Detta ger i sin tur det färgtecken MID$ plockar ut och stoppar in i B$.

A$ = " ■ E £ ◣ ◨ ↑ ← π "

Den fjärde raden är:

```
4 PRINT B$ "  ";
```

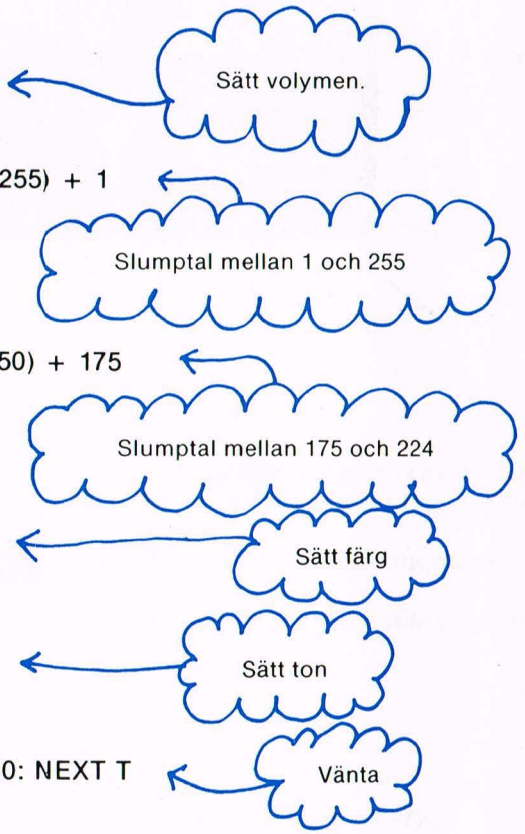
Denna rad talar om för VIC att använda den färg som finns i B$ och rita fåglarna med. Kommandot PRINT B$, ger samma effekt som om du skulle tryckt på motsvarande färgtangent. Men när vi använder B$ kan färgtecknet ändras automatiskt. När N ändras, ändras också B$ och därmed färgen på fåglarna.



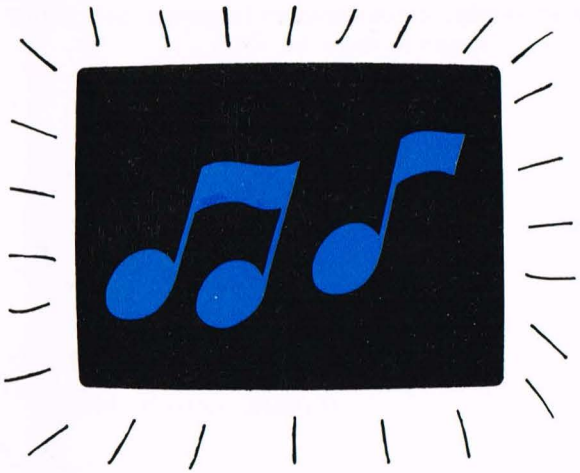
VIC's möjligheter att generera slumptal kan användas för färg, ljud eller grafiska konstruktioner. Låt oss åter besöka VIC's färgshow, men nu med slumptal och lite brus därtill.

I det sista exemplet producerades färgerna och ljudet slumpmässigt. Skriv in följande rader till VIC:

```
NEW

1POKE 36878, 3

2C = INT(RND(1)*255) + 1

3S = INT(RND(1)*50) + 175

4POKE 36879, C

5POKE 36875, S

6FOR T = 1 TO 100: NEXT T

7GOTO 2
```



Pratbubblorna förklarar vad som görs i programmet. I kapitel 5 förklaras mer om musik. När du är klar att köra exemplet, rensa då skärmen och skriv [R] [U] [N] och tryck **RETURN** .



Oj oj! Vilka knepiga kombinationer av färg och musik. Du kan inte vänta dig att slumpmusik skall tillfredsställa alla smakriktningar.

Vill du stoppa detta fyrverkeri så tryck på RUN STOP. Sedan skriver du dessa två rader för att få tyst på VIC och få tillbaks normala färger:

```
POKE 36875, 0
```

Stäng ljudet.

```
POKE 36879, 27
```

Återställ färgen.

Det är dags för dig att göra dina egna experiment med färg och ljud. Upplev din färgglada och musikaliska dator.

GRAFIK PÅ TANGENTBORDET

En av de speciella möjligheter som VIC erbjuder är en uppsättning med grafiska tecken. De flesta av tangenterna har två stycken grafiska tecken tryckta på framsidan. Du kan få dessa grafiska tecken utskrivna på skärmen eller om du har en Commodore-skrivare kan tecknen skrivas ut på papper.

För att skriva ut det grafiska tecknet på vänstra sidan håller du Commodoretangenten nedtryckt och trycker på tangenten med det grafiska tecknet. Den högra sidans grafik får du om SHIFT tangenten hålls nedtryckt och därefter grafiktangenten.

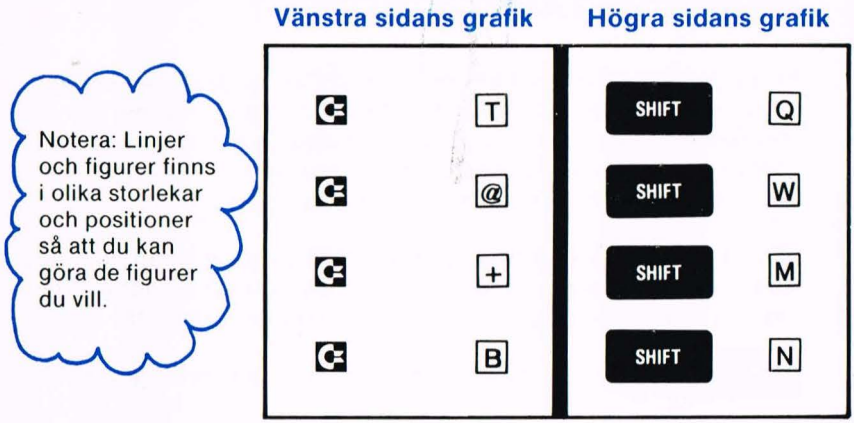
VIC TIPS

Observera: Om du trycker på SHIFT och Commodore-tangenten samtidigt byts teckenuppsättning. En del av de grafiska tecknen byts mot små bokstäver. När små bokstäver används kan endast vänstra sidans grafisk användas. För att få tillbaka all grafik tryck på SHIFT och C= samtidigt.

Det enklaste sättet att använda VIC's grafik är att hålla COMMODORE C= eller SHIFT tangenten nedtryckt, och trycka på den grafiska symbol som du vill få utskriven på skärmen. Exempel, skriv:

SHIFT S

Du ser ett blått hjärta på skärmen. Försök med några andra tangenter, exempelvis dessa:

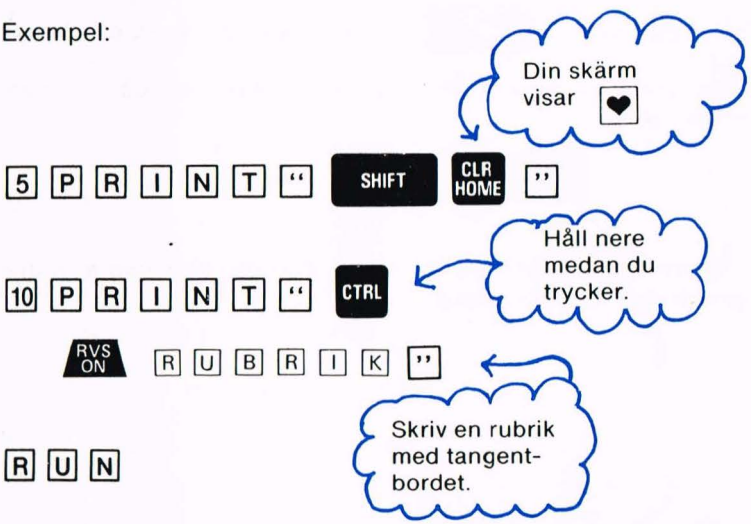


Notera att den vänstra sidans grafik är bra för att skapa tabeller och formulär. Högra sidan är bra när du vill göra illustrationer, spelkort osv.

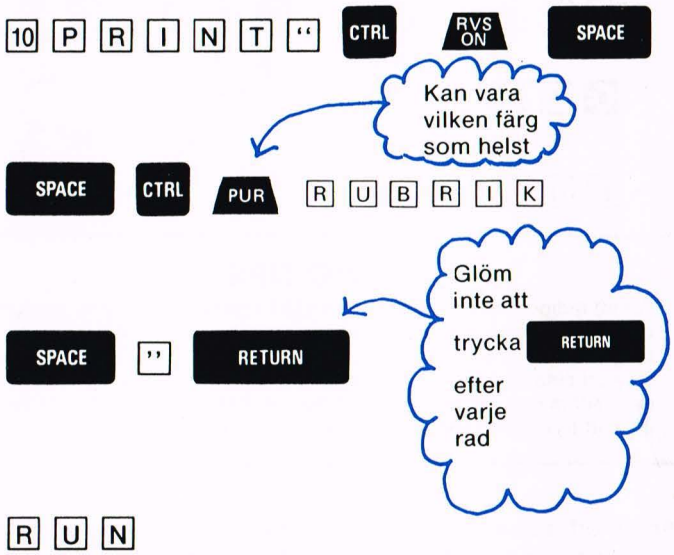
GRAFIK I RUBRIKER

Grafiken kan användas till mycket annat än spel. Grafiken hjälper till att framhäva en viss text eller rubrik. Det finns många olika sätt. Det enklaste är att skriva ut texten reverserat. Tryck endast på CTRL , RVS ON och skriv den text du vill.

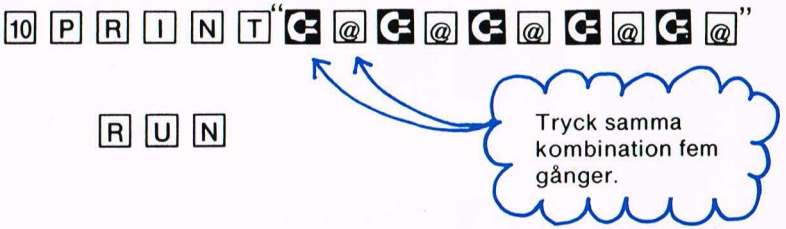
Exempel:



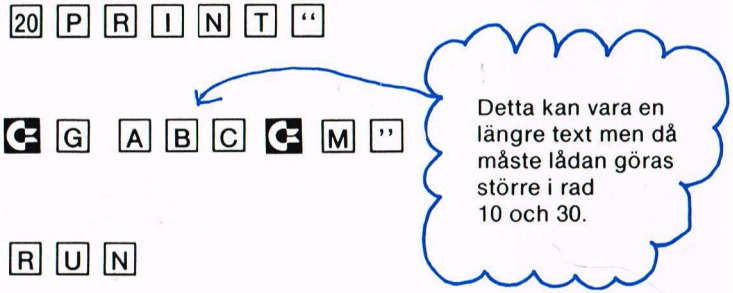
Nu försöker vi göra en rubrik med färg...



En annan metod är att rita en låda kring ordet. Du använder de grafiska tecknen för att åstadkomma detta. Var noga med att göra det precis som det står. Gör det steg för steg.



Skärmen visar en rak linje. På nästa rad finns din text och de vertikala linjerna på varje sida av texten för att skapa din låda.



Nu har vi skrivit texten. Då lägger vi dit undersidan på lådan.

30 P R I N T " C T C T C T

C T C T "

5 gånger!

R U N

VIC TIPS

För att redigera ditt program, skriv LIST och tryck RETURN. Nu kan du gå tillbaks och rätta i programmet genom att flytta markören med CRSR-tangenterna och ändra de felaktiga raderna med INST/DEL-tangenten. När du gjort dina ändringr på raden trycker du RETURN för att VIC skall ändra den i sitt minne. Du kan också skriva om hela raden när du vill. Kom bara ihåg att trycka RETURN när du skrivit färdigt.

Här är ett annat sätt att dra uppmärksamheten till en text. Följande text kommer att blinka ett antal gånger på din skärm...

N E W

Skriv detta för att radera programmet från VIC's minne.

10 FOR H = 1 to 250

20PRINT " SHIFT CLR HOME R U B R I K "

25FOR T = 1 TO 50: NEXT

30PRINT " SHIFT CLR HOME CTRL RVS ON

R U B R I K "

35FOR T = 1 TO 50: NEXT

40NEXT H

R U N

Överlappar inte texterna varandra, ändra på texten inom citationstecknen. För att få den att blinka långsammare eller snabbare, ändra i talen i raden 25 och 35.

Figur-ritning

- Flygande fåglar
- Studsande bollar
- Kontroll av markör
- POKE och PEEK

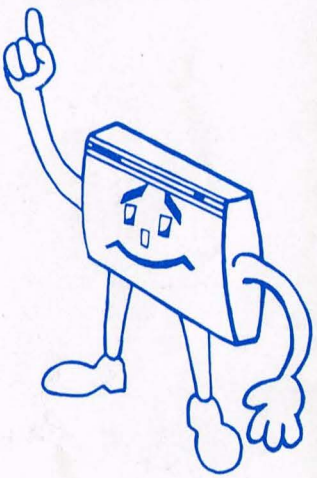
Skriv detta program:

Skriv programmet precis som rutan nedanför visar, och se vad som händer!

```
10 PRINT " CLR HOME ";
20 PRINT " ▨ ○ ▨ "
30 PRINT " □ ▩ □ "
40 PRINT " ▨ □ ▨ "
50 FOR T=1 TO 300: NEXT
60 PRINT " CLR HOME ";
70 PRINT " □ ○ □ "
80 PRINT " ▨ ▩ ▨ "
90 PRINT " □ ◧ ◧ "
100 FOR T=1 TO 300: NEXT
110 GOTO 10
```

Skriv RUN och tryck RETURN

För att stoppa programmet, tryck RUN STOP tangenten.



FLYGANDE FÅGLAR

Detta kapitel beskriver hur du kan använda VIC's grafiska möjligheter för att skapa illusionen att figurer i skärmen rör på sig.

Denna illusion att figurer rör sig kallar vi för animering. Animering kan göra alla typer av program, från spel till administrativa, roliga och spännande.

Låt oss börja med att skriva in följande rader:

Tryck ned SHIFT tangenten och tryck på CLR HOME tangenten.

N E W och tryck RETURN

1 SPACE P R I N T " ♥ ◡● "

Håll SHIFT och tryck CLR HOME

För att få fåglar

	HÅLL	TRYCK
◡●	SHIFT	J Q K
●◠	SHIFT	U Q I

För att veta mera se kap. 3 i "Grafik"

och tryck RETURN .

2 SPACE F O R SPACE T =

1 T 0 1 5 0 : N E X T T ← Vänta ett tag

och tryck RETURN .

3 SPACE P R I N T " ♥ ●◠ " ← Se rad 1

och tryck RETURN .

4 SPACE F O R SPACE T =

1 T 0 1 5 0 : N E X T T ← Vänta igen

och tryck RETURN .

5 SPACE G O T O SPACE 1

Vad du skrev nyss bör se ut så här:

```
1 PRINT " ♥ ◡●◡ ";
2 FOR T=1 TO 150: NEXT T
3 PRINT " ♥ ◜●◝ "
4 FOR T=1 TO 150: NEXT T
5 GOTO 1
```

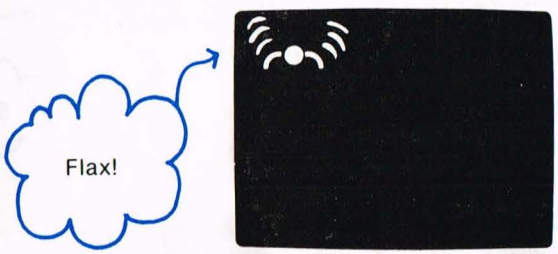
Titta på vad du skrivit. Stämmer allt överens med bilden ovanför? Om det inte gör det skriv då om de rader som är felaktiga. När allt är OK skriv då:

R U N och tryck RETURN

TIPS: Du kan ända en rad genom att skriva in den en gång till eller ändra direkt på raden. Den sista ändringen gäller alltid. Kom ihåg att trycka RETURN när du skrivit eller ändrat färdigt raden. Vill du radera raden behöver du bara skriva radens nummer och trycka

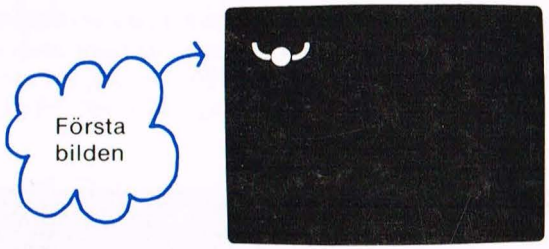
RETURN

Betrakta din första animering med VIC. Det ser ut som fågeln flaxar i vänstra hörnet på skärmen. Den kämpar på bra men tycks ha en hård motvind. Den kommer ingenstans.



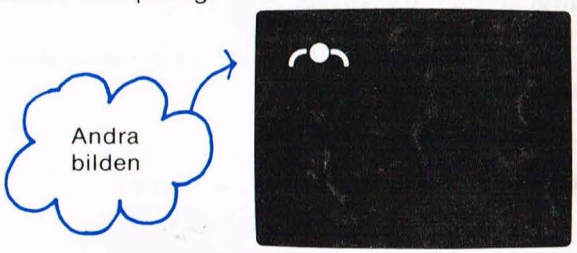
Du ser hur fågeln faktiskt ser ut att flaxa med sina vingar. Studera detta exempel en stund. Hemligheten bakom all animering finns beskrivet i detta exempel. Hur du skapar illusionen av att någonting rör sig beskrivs i några få enkla steg. Här följer en summering hur du kan få en figur eller ett tecken att röra på sig, blinka, flaxa osv.

STEG 1. Skriv ut tecknet eller teckenkombinationen på skärmen.



STEG 2. Vänta ett litet tag. Det gör du med en FOR NEXT loop. Exempel på tidsfördröjningar visas i rad 2 och 4 i programmet med de flygande fåglarna. Dessa rader talar om för VIC att den skall räkna från 1 till 150. VIC kan räkna till 150 snabbt. Försök att ändra 150 till 500 (eller ett annat nummer) och se vad som händer.

STEG 3. Skriv ut tecknet igen men med någon ändring gjord. Beroende på den tidigare tidsfördröjningen tror dina ögon att figuren har rört på sig.



STEG 4. Vänta igen. Beroende på vilken effekt du vill ha kan denna tidsfördröjning vara längre eller kortare än den första. I detta exempel är de båda fördröjningarna lika långa.

STEG 5. Repetera de fyra första stegen som vi nyss har beskrivit. Detta åstadkoms genom GOTO-kommandot som talar om för VIC att hoppa tillbaks till den första raden och börja om från början. Programmet skriver ut fågeln med vingarna uppåt, räknar till 150 sedan skriver den ut fågeln med vingarna nedåt, räknar till 150 igen och hoppar sedan tillbaka till början igen.

I det första och tredje steget där tecknen skrivs ut är det viktigt att försäkra sig om att inga gamla delar av figuren lämnas kvar på skärmen. I vårt exempel är det inget problem eftersom skärmen rensas varje gång. (SHIFT och CLR HOME i varje PRINT-kommando rensar skärmen och flyttar markören längst upp till vänster och varje version av fågeln (vingarna upp och vingarna ner) skrivs ut i övre hörnet.

Du vill kanske experimentera med dessa exempel.

Tryck RUN STOP för att vingarna skall sluta flaxa. Lista ut programmet och ändra värdet 150 i rad 2 och 4. Försök att bara ändra ett av värdera och lämna det andra kvar på 150. Försök att stoppa in värdet 50 på båda raderna. Flyger din fågel fortare? Försök med 500? 1000?

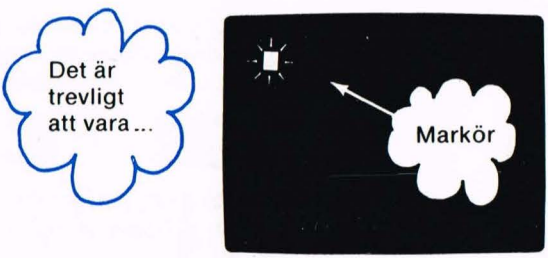
Hmmm... fågeln verkar ganska slö...

Du vet redan att du kan rensa skärmen genom att hålla SHIFT nedtryckt och trycka på CLR HOME tangenten. Från programmet kan du tala om för VIC att den skall rensa skärmen. För att göra så skriver du:

PRINT " ♥ ";

Detta tecken syns när du håller nere SHIFT och trycker CLR HOME

När du skall skriva ut meddelandet och trycker på de två tangenterna (SHIFT och CLR HOME) som du gör för att rensa skärmen, kommer det fram ett reverserat hjärta i utskriftsfältet. Denna symbol är en signal till VIC att när den skriver ut meddelandet skall den rensa skärmen och placera markören i HOME-positionen. HOME-positionen är det övre vänstra hörnet på skärmen.



Försök att skriva PRINT kommandot direkt. Rensas skärmen?

Skriv: PRINT " ♥ ";

Ja!
Fungerar i
direktmode

Fyll skärmen med text och grafiska tecken. När din skärm är tillräckligt full skriv då följande:

PRINT " S ";

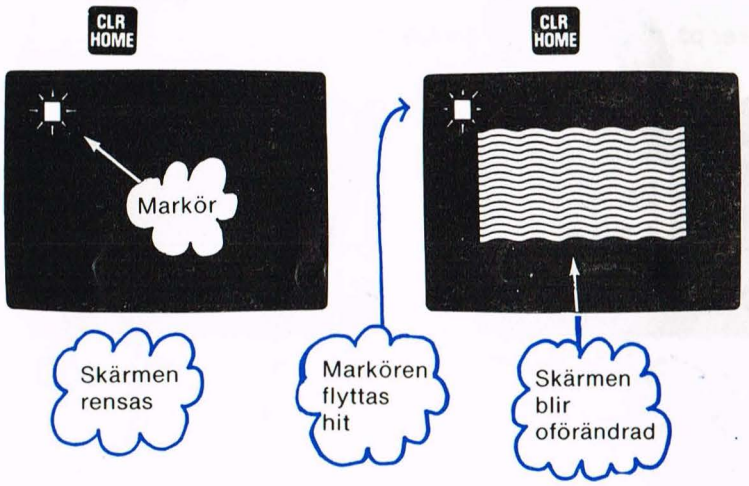
Håll inte SHIFT nere.
Tryck bara CLR HOME

Noterade du skillnaden från det tidigare PRINT kommandot? Denna gång flyttade sig markören bara upp till HOME-positonen på skärmen. Skärmen rensades inte. Om du inte håller

SHIFT nedtryckt när du trycker på i PRINT-kommandot

skrivs ett reverserat S ut istället för ett hjärta. Det reverserade S'et talar om för VIC att flytta markören upp till HOME-positionen, men inte att rensa skärmen.

Skriv: SHIFT CLR HOME



STUDSANDE BOLL

Rensa skärmen och skriv in följande till VIC:

(Tryck på RETURN när du skrivit färdigt varje rad.)

NEW
1 PRINT " ● " ;

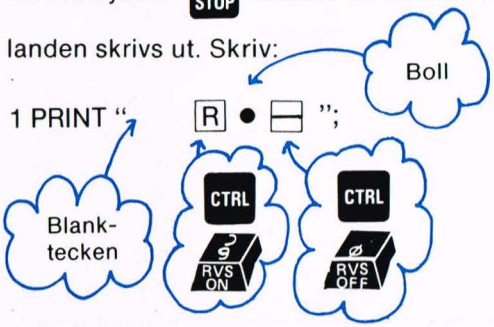
SHIFT Q
Blank-tecken

2 GOTO 1

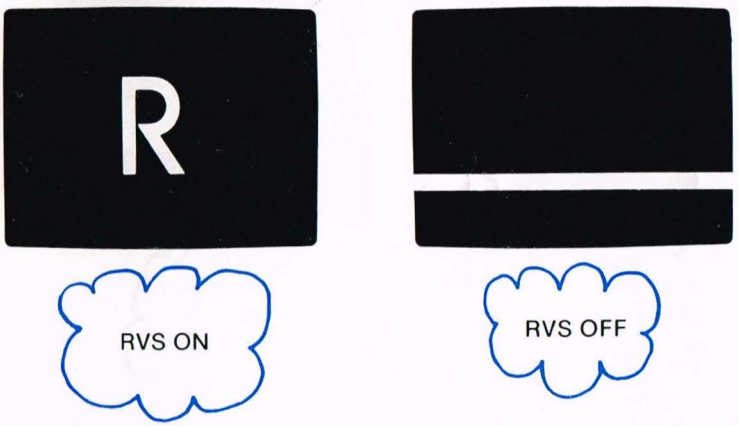
RUN

När du skrivit RUN och tryckt RETURN kommer skärmen att fyllas med blå bollar. Låt bollarna flyta omkring ett tag och tryck sedan RUN STOP .

Om du trycker RUN STOP orsakar detta att BREAK och READY meddelanden skrivs ut. Skriv:



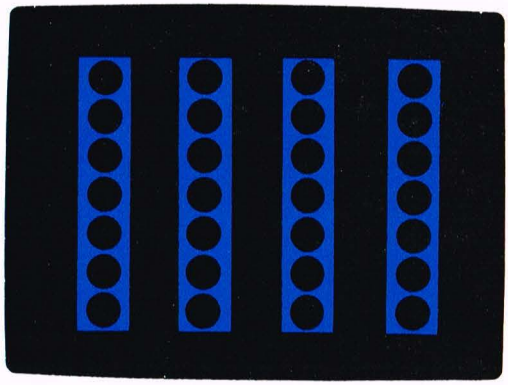
De två reverserade tecknen är symboler som VIC lägger in i meddelandefältet när du håller CTRL tangenten nedtryckt och trycker på 9 RVS ON och 0 RVS OFF tangenterna.



När du skrivit in den nya raden skriv då RUN till VIC så att den kör igång programmet. Skriv:

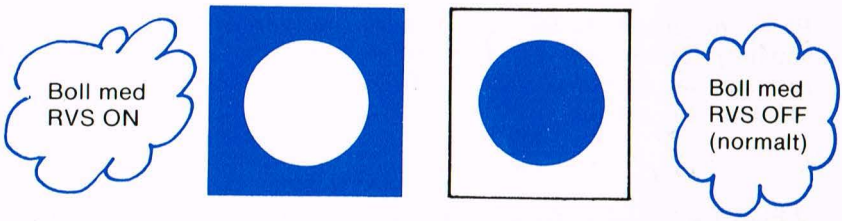
RUN

Vad händer med bollarna? Tryck på CTRL tangenten för att sakta ned utskriften. Nu är det kolumner med blå rektanglar med hål i. Kan du gissa varför?



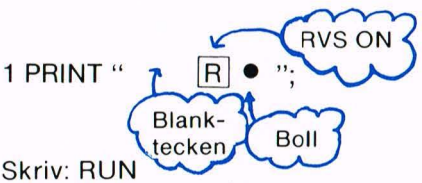
Blå linjer med vita hål

Kontrolltecknet RVS ON före bollsymbolen gör att VIC reverserar färgerna som bollarna visas med. Bollarna är normalt blå och omges av en vit bakgrund. Reverseringen gör bakgrunden blå och bollarna vita.



Kontrolltecknet RVS OFF efter bollsymbolen talar om för VIC att återgå till normal tecken- och bakgrundsfärg. Det är som att reversera det reverserade. Du slutar där du startade. Om inte RVS

OFF används får du en intressant effekt. Tryck RUN STOP och skriv in följande rad.



Skriv: RUN

När RVS OFF inte används, blir alla mellanrum på skärmen reverserade och hela skärmen blir en gemensam ruta. Vilken intressant mönsterskapare VIC kan bli. Varför inte använda VIC till design av textilmönster, tapeter och golv? Vad tror du att du kan göra med VIC?

MARKÖRTANGENTERNA

De fyra markörtangenterna tillåter dig att flytta markören vart du vill på skärmen. Du kan lägga in markörtangenterna i PRINT-utskrifterna för att hjälpa till att få dina texter på rätt plats. Här är fågelexemplet från början av kapitlet med några extra markörtangenter instoppade.

Rensa skärmen och skriv in följande rader:

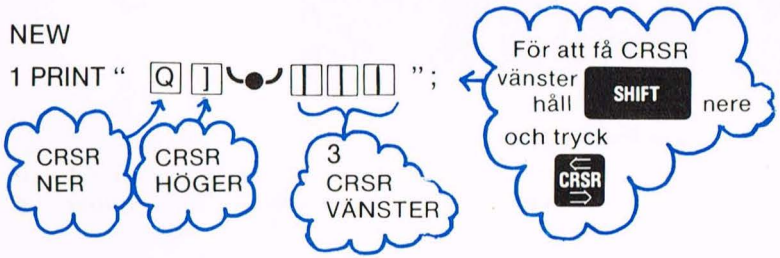
```
NEW
1 PRINT " Q ] ◡●◡ ▯▯▯ ";
```

CRSR NER

CRSR HÖGER

3 CRSR VÄNSTER

För att få CRSR vänster håll SHIFT nere och tryck CRSR



På denna rad använder vi flera markörtangenter (nedåt, höger, vänster) för att animera fågelns rörelser. Du kommer att se rörelsen när du kör programmet. Denna rad innehåller markör "vänster" tre gånger.

```
2 FOR T=1 TO 150: NEXT T
3 PRINT " ◠●◠ ▯▯▯ ";
```

Vänta

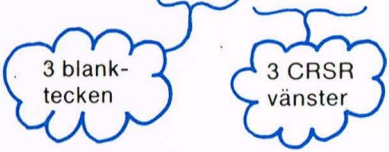
3 CRSR vänster

Denna rad innehåller tre stycken Markör vänster.

```
4 FOR T=1 TO 150: NEXT T
5 PRINT "    ▯▯▯ ";
6 FOR T=1 TO 150: NEXT T
7 GOTO 1
```

Fördröjning

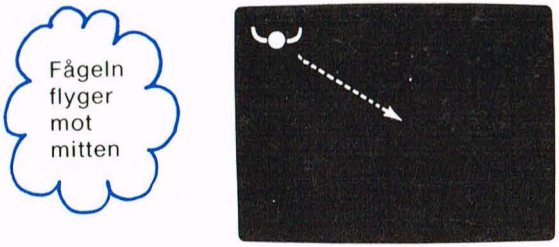
Radera fågel



3 blank-tecken

3 CRSR vänster

Vänta igen



När du skrivit in exemplet och kontrollerat att du skrivit rätt, skriver du RUN och tittar på fågeln. Markörtangenterna i detta exempel flyttar fågeln nedåt över skärmen. Enkelt VA!

Din tur igen. Experimentera med exemplet och skapa en annan figur som rör sig över skärmen.

ANIMERING MED POKE OCH PEEK

Hittills har vi lärt oss att använda PRINT-kommandot tillsammans med markörkontroll och färgkontroll för att flytta figurer runt i skärmen. Detta är troligtvis det snabbaste sättet att flytta figurer i BASIC, men det finns ett som är bättre. I de flesta spel måste objektet som rör sig känna om det finns andra objekt i närheten på skärmen. Den enda praktiska vägen att göra detta är att använda PEEK och POKE kommandona.

För att förstå denna teknik måste du först förstå hur video-skärmen arbetar. Varje position i skärmen representeras av en minnesposition i VIC's minne (RAM). Eftersom det finns 506 positioner på skärmen så finns det 506 positioner i minnet för att få plats med alla tecken. Eftersom varje minnesposition i minnet kan innehålla ett värde från 0 till 255 finns det 256 möjliga värden som varje minnesposition kan innehålla. Dessa är de 256 olika tecken som VIC kan presentera (se bilaga H). Ett nummer i en position i skärmminnet är koden för ett tecken i den positionen. Skärmminnet i VIC börjar normalt i position 7680, och slutar i position 8185. Position 7680 är positionen i skärmens övre vänstra hörn. Position 7681 är positionen ett steg till höger om den första osv till slutet på raden. Positionen efter den sista i varje rad är det första tecknet i nästa rad.

Låt oss anta att vi håller på att studsa en boll på skärmen. Bollen befinner sig mitt i skärmen, kolumn 10 och rad 10. (Övre vänstra hörnet är kolumn 0 och rad 0.) För att räkna ut minnespositionen kan denna formel användas.

P = 7680 + X + 22*Y

där X är raden och Y kolumnen. Minnespositionen till bollen är 7680 + 10 + 220, eller 7910. Rensa skärmen och skriv:

POKE 36879,8

Detta ändrar färgen på skärmen till svart. Skriv:

POKE 7910,81

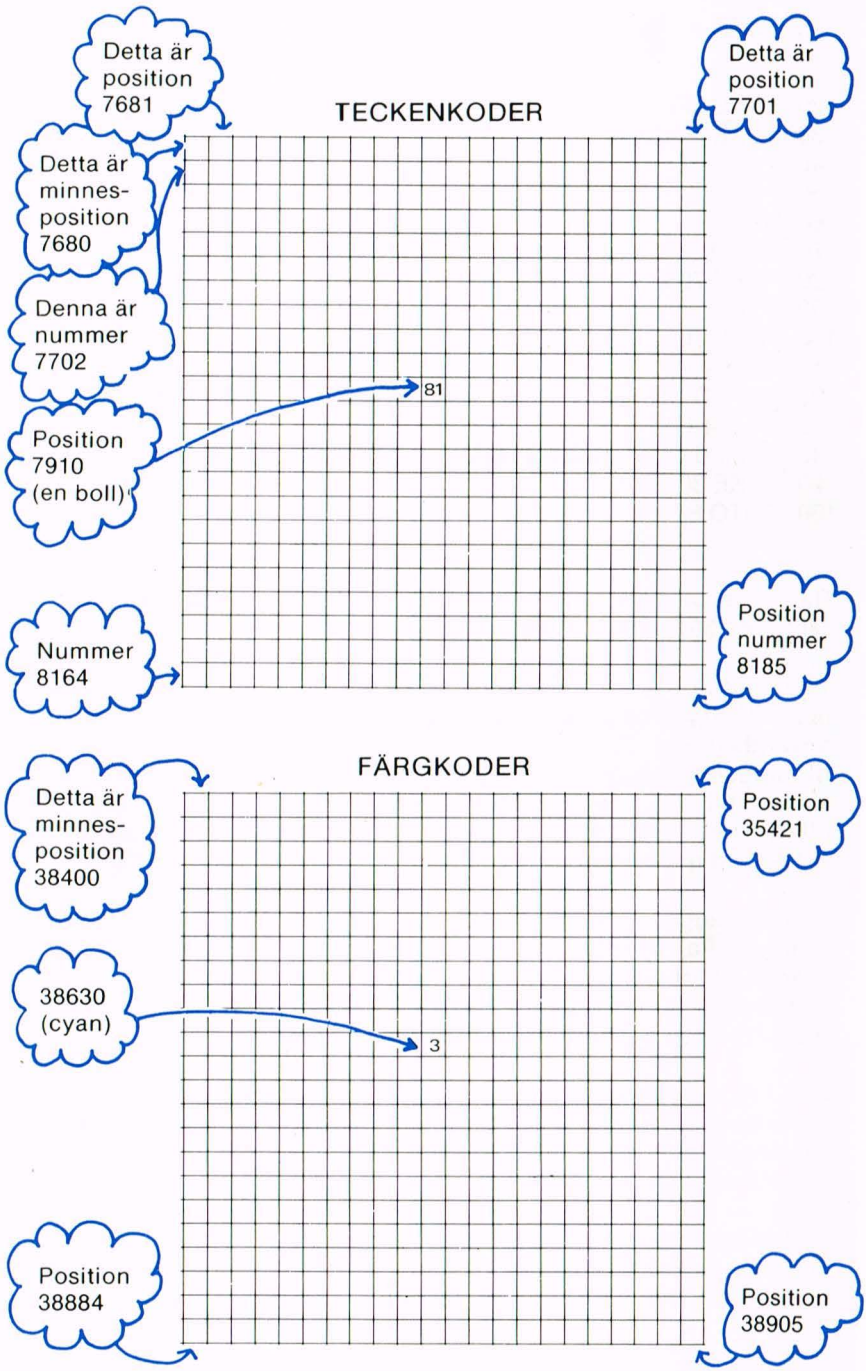
En boll dyker upp mitt på skärmen. Du har placerat tecknet direkt på skärmen utan att använda kommandot PRINT.

Bollen på skärmen är vit. Det finns ett sätt att ändra färgen på varje teckenposition i skärmen genom att använda POKE. Skriv:

POKE 38630,3

Bollens färg ändras till cyan. För varje position i VIC's skärm finns det två minnespositioner, en för teckenkoden och en för färgkoden. Färgminnet börjar i position 38400 (övre vänstra hörnet), och fortsätter 506 positioner framåt. Färgkoden är från 0 till 7 vilket ger 8 färger numrerat från 1 till 8 på tangentbordet. (Andra kommer att ge konstiga resultat. Se VIC's programmeringshandbok som ger en komplett beskrivning).

SKÄRMMINNESKARTA



Här följer ett kort program för en boll som studsar runt i skärmen, följd av en detaljerad förklaring.

```
10 PRINT " SHIFT CLR HOME "
20 POKE 36879,9
30 POKE 36878,15
40 X=1
50 Y=1
60 DX=1
70 DY=1
80 POKE 7680+X+22*Y, 81
90 FOR T=1 TO 10: NEXT
100 POKE 7680+X+22*Y, 32
110 X=X+DX
120 IF X=0 OR X=21 THEN DX=-DX: POKE 36876, 220
130 Y=Y+DY
140 IF Y=0 OR Y=22 THEN DY=-DY: POKE 36786, 230
150 POKE 36876, 0
160 GOTO 80
```

Rad 10 rensar skärmen, och rad 20 gör skärmen svart med en vit ram. Rad 30 ger maxvolym på tongeneratorn.

Variablerna X och Y används i rad 40 och 50 för att hålla reda på den tillfälliga kolumnen och raden för bollen. DX och DY som används i rad 60 och 70 är den vertikala och horisontala riktningen på bollens rörelse. När X ökas med +1 flyttas bollen en kolumn till höger. När X minskas med —1 flytts bollen en kolumn till vänster. Addering av +1 och Y flyttar bollen en rad nedåt och —1 en rad uppåt.

Rad 80 lägger in bollens tecken i den tillfälliga positionen i skärmen. Rad 90 är en tidsfördröjning, för att lämna kvar bollen på skärmen tillräckligt länge för att ditt öga ska uppfatta den. Rad 100 raderar nu bort bollen genom att skriva dit ett blanktecken (kod 32) i den position där den fanns på skärmen.

Rad 110 adderar riktningsfaktorn till X. Rad 120 testar för att se om bollen nått till kanten på skärmen. Om det är så reverserar den riktningen och sätter på tongeneratorn. Rad 130 och 140 gör samma sak fast med Y-värdet.

Rad 150 stänger av tongeneratorn, om den blev påslagen, för att avsluta studseffekten. Rad 160 hoppar tillbaks för att skriva ut bollen igen, men den här gången i en annan position.

Lek litet med detta program. Genom att ändra värdet i rad 80 från 81 till en annan teckenkod, kan du ändra bollen till någon annan figur. Om du ändrar någon av DX eller DY till 0, kommer bollen att hoppa rakt upp och ned eller i sidled istället för diagonalt.

Nu kan vi stoppa in litet intelligens i detta program. Hittills har vi kontrollerat om X- och Y-värdena kommer ut till skärmens kanter. Addera följande rader till programmet som vi tidigare visat. (Skriv bara in dessa rader och de adderas automatiskt.)

```
 32 FOR L = 1 TO 10
 35 POKE 7680 + INT (RND (1) *506), 102
 37 NEXT
155 IF PEEK (7680 + X + 22*Y) = 102 THEN DX = -DX: DY = -DY:
     POKE 36876, 180: GOTO 110
```

Rad 32 till 37 lägger in 10 stycken grå tecken på skärmen i slumpmässiga positioner. Rad 155 kollar (PEEKar) för att se om bollen håller på att hoppa in i någon av dessa grå tecken, och reverserar bollens riktning om så är fallet.

Se bilaga H och I för mera information om animering.

5

Ljud och musik

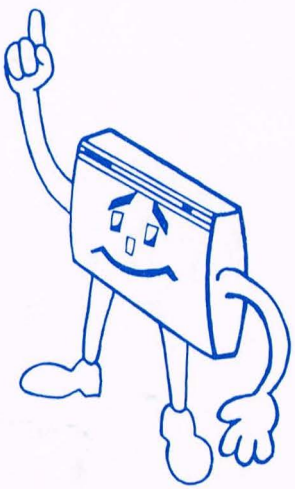
- Att göra musik
- VIC's fyra röster
- Brusgenerator
- VIC som ett piano
- Spela melodier
- Några ord om POKE

Skriv detta program:

Skriv programmet precis som rutan nedanför visar, och se vad som händer!

```
NEW
 10 PRINT " SHIFT CLR HOME "
 20 FOR I = 1 TO 5
 30 POKE 36873 + I, 0
 40 NEXT I
 50 PRINT "VILKEN STÄMMA (1-4)?"
 60 PRINT "OM FÄRDIG, TRYCK 0"
 70 INPUT N
 80 IF N = 0 THEN END
 90 PRINT "VILKEN TON (128-254)"
100 INPUT P
110 POKE 36878, 4
120 POKE 36873 + N, P
130 FOR J = 1 TO 2000: NEXT N
140 GOTO 10
```

För att stoppa programmet, tryck RUN STOP tangenten.



MUSIK

Du har kanske upptäckt att du och din VIC kan skapa musik och ljudeffekter! Detta kapitel är en introduktion till de musik- och ljudeffekter som VIC kan skapa. Du kommer att lära dig hur du kontrollerar dessa ljud och kan spela all musik från Bach till Rock. Så ställ en ljusstake på din TV, och var beredd på en konsert!

Låt oss först ta reda på hur du får fram en ton:

Håll nere SHIFT tangenten och tryck på CLR HOME tangenten.

N E W och tryck på RETURN

P O K E SPACE 3 6 8 7 8 , 3

och tryck RETURN

P O K E SPACE 3 6 8 7 5 , 2 2 5

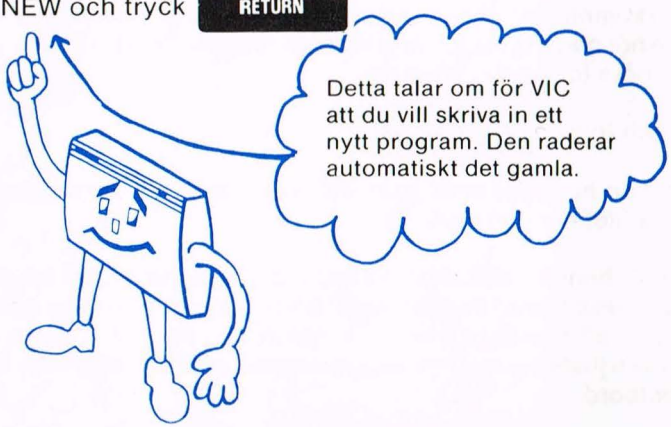
och tryck RETURN

Du borde nu höra en ton från TV-apparatens högtalare. Om inte, försök att justera volymkontrollen på din TV. Ställ in den så att du får en lagom nivå. Du har just nu spelat ett normal-C. VIC:s repertoar består av 128 toner. Stäng av tonen genom att skriva:

P O K E SPACE 3 6 8 7 5 , 0

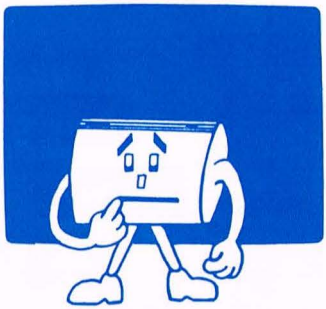
Låt oss ta reda på hur många toner VIC kan spela:

Skriv NEW och tryck RETURN



Skriv detta program:

```
1 POKE 36878, 15 RETURN
2 FOR I = 128 TO 255 RETURN
3 POKE 36875, I RETURN
4 FOR D = 1 TO 100 RETURN
5 NEXT D RETURN
6 NEXT I RETURN
7 POKE 36875, 0 RETURN
```



Titta efter om VIC's skärm ser ut som den i boken. Om någon rad är felaktig, skriv bara om den en gång till. Var bara säker på att radnumret kommer med. VIC kommer automatiskt att byta ut den gamla raden mot den nya. För att få en listning av raderna i ditt program skriv:

LIST och tryck RETURN

Om alla rader är OK, skriv:

RUN och tryck RETURN

och du kommer att höra alla de 128 toner som VIC kan spela. (Om du inte hör någonting, kontrollera om raderna är rätt skrivna.) Om du vill höra tonerna igen skriv:

RUN och tryck RETURN

Nu har du hört de toner som VIC kan spela. Vi kommer nu att förklara litet hur detta går till.

Inuti VIC, finns en krets som kallas Video Interface Chip (som har gett VIC dess namn) som handhar både ljud och bild. Den ger VIC möjlighet att sända ut ljud till högtalaren i din TV. Volymen på tonen kan justeras med en volymkontroll på TV:n eller från VIC's tangentbord.

VIC'S FYRA STÄMMOR

Din VIC har fyra röster... dvs den spelar i fyra stycken olika stämmor samtidigt! Du kan tänka dig stämmorna som en sopran, alt, tenor och en brusstämma. Varje stämma har sitt eget kontrollnummer. Genom att använda detta nummer kan vi slå på stämman för att spela en ton eller göra en ljudeffekt. Vi använder kommandot POKE för att göra detta.

VIC's fyra stämmor har följande nummer:

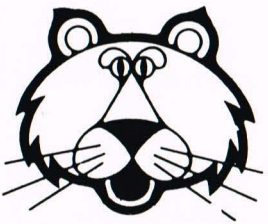
36874 (stämma 1 - musik)
36875 (stämma 2 - musik)
36876 (stämma 3 - musik), och
36877 (stämma 4 - brus).

Volymkontrollen har nummer 36878.

36874 . . . (bas)

36875 . . . (tenor)

36876 . . . (sopran)



36877 . . . (brus)
stämma 1

36878 . . . (volym)

För att spela en ton, skriv POKE och stämmans nummer, ett kommatecken, och ett nummer som representerar tonen du vill spela.

T ex skriv:

POKE 36874,128 och tryck RETURN

Du har just nu spelat ton #128 genom att använda stämma 1 med nummer 36874, den första låga stämman. Från och med nu kallar vi denna stämma för S1 (förkortning för stämma 1), den andra S2, den tredje S3 och den fjärde S3. S2 sjunger högre toner än S1 och S3 högre än S2. S4 spelar med ett brus i sin röst... denna röst skapar "vitt brus" för ljudeffekter.

För att stänga av en stämma läggs värdet 0 in:

POKE 36874, 0 and press RETURN

Detta kanske verkar aningen komplicerat för dig som inte är en "datormusiker", så vi har gjort det litet lättare för dig. Du kan spara mycket tid genom att skriva följande korta program för att översätta stämmornas nummer till kortare nummer.

Skriv in till din VIC precis som visas:

```
NEW
S1 = 36874      RETURN
S2 = 36875      RETURN
S3 = 36876      RETURN
S4 = 36877      RETURN
V  = 36878      RETURN
```

Detta tillåter oss att referera till stämmornas nummer som förkortningarna S1, S2, S3 och S4 och V (volym). Innan vi fortsätter kontrollerar du att varje rad är rätt inskriven. Nu är du klar för att göra ljudeffekter den enkla vägen!

Denna rad stoppar in 10 till positionen där volymkontrollen finns. Volymkontrollen kan lagra ett värde från 0 till 15, ju högre nummer desto högre volym.

```
POKE S1, 195
POKE S2, 215
POKE S3, 231
```

Experimentera litet grand med denna metod att producera ljud. Det är mycket enkelt eller hur? När du vill stoppa ljudet (för at låta dina öron vila ett tag), lägger du in värdet 0 i stämmorna. Så här:

```
POKE S1, 0
POKE S2, 0
POKE S3, 0
```

Här är en tabell över de värden som kan läggas in i stämmorna för få olika toner: (OBS: Värden under 128 ger ingen ton.)

TABELL ÖVER MUSIKTONER

TON	VÄRDE	TON	VÄRDE
C	135	G	215
C#	143	G#	217
D	147	A	219
D#	151	A#	221
E	159	B	223
F	163	C	225
F#	167	C#	227
G	175	D	228
G#	179	D#	229
A	183	E	231
A#	187	F	232
B	191	F#	233
C	195	G	235
C#	199	G#	236
D	201	A	237
D#	203	A#	238
E	207	B	239
F	209	C	240
F#	212	C#	241

Genom att stoppa in dessa värden i de tre första stämmorna (S1—S3), kan du spela melodier. (Vitt brus är inte speciellt

lämpligt här... om du inte drömmer om att få spela första datorsolist med Pink Floyd!) Tyvärr är den här processen med att använda POKE väldigt långsam för en människa... så låt oss lägga in POKE-kommandona i ett program och låta datorn göra arbetet! Vill någon höra "Humlans flykt"? Eller kanske "Maple Leaf Rag"?

DEN VITA BRUSGENERATORN

Den fjärde stämman, S4, har numret 36877. Vi har kallat den för "brus" därför att den egentligen är en vitt-brusgenerator som används för specialeffekter. Försök att härma ett flygplan:

```
POKE V, 6
POKE S4, 130
```

En väldigt låg ton

Ett fyrtmotorigt propellerplan, eller hur? Nu försöker vi med en vind som slår mot vingarna på ett segelflygplan?

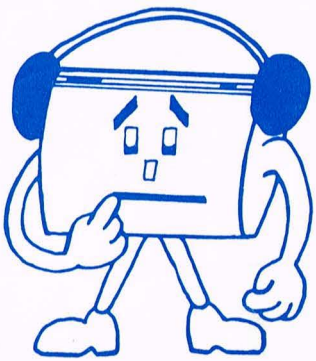
```
POKE S4, 240
```

Glöm inte att sätta på volymkontrollen med:

```
POKE V,4
```

eller vad du vill sätta volymen till. Stäng av ljudet med:

```
POKE S4, 0
```



Försök byta ut DATAsatserna för att spela andra melodier. Här är till exempel en gammal favorit:

```
300 DATA 225, 360, 225, 360, 225, 240
310 DATA 228, 120, 231, 360, 231, 240
320 DATA 228, 120, 231, 240, 232, 120
330 DATA 235, 720, 240, 360, 235, 360
340 DATA 231, 360, 225, 360, 235, 240
350 DATA 232, 120, 231, 240, 228, 120
360 DATA 225, 480
370 DATA -1
```

Eller om du föredrar klassiskt:

```
300 DATA 217, 400, 213, 400, 223, 400
310 DATA 227, 200, 234, 200, 230, 400
320 DATA 227, 200, 234, 200, 230, 400
330 DATA 223, 400, 227, 400, 217, 400
340 DATA 213, 600, -1
```

Använd så många rader som du vill för att spela längre melodier, men för syftet med detta exempel använd bara 3 toner per rad.

VIC SOM PIANO

Till slut, här är ett program som gör att du kan använda VIC:s tangentbord som ett piano.

```
NEW

 10 REM  LAGRA LJUDREGISTER
 20 S2 = 36875
 30 V  = 36878
 40 POKE S2, 0

100 REM  LÄS B-DUR SKALAN
110 FOR N = 1 TO 8
120 READ A (N)
130 NEXT N

140 DATA 223, 227, 230
150 DATA 231, 234, 236
160 DATA 238, 239

200 REM  SPELA PÅ TANGENTBORDET
210 POKE V, 3

220 GET A$: IF A$ = " " THEN 220
230 N = VAL (A$)

240 IF N = 0 OR N = 9 THEN 300
```

Förkortar tonregistren och stänger av dem.

Läser B-durs tonskala från raderna 140-160

Innehåller tonvärdet för B-dur

Sätter på volymen

Tar reda på vilken tangent som är tryckt.

Avslutar programmet om du trycker 0 eller 9

```
250 POKE S2, 0
260 FOR T = 1 TO 25: NEXT T

270 POKE S2, A (N)
280 GOTO 220

300 REM  SLUT MODUL
310 POKE S2, 0
```

Ett kort avbrott mellan tonerna.

Spelar tonen och hoppar till-baka för att ta nästa

Stänger av ljudet

När du nu skriver RUN (och trycker RETURN) kan du spela melodier på din VIC. Siffertangenterna i den översta raden ger de olika tonerna:

```
**********************************************************
  1     2     3     4     5     6     7     8
 DO    RE    MI    FA    SO    LA    TI    DO
**********************************************************
```

VIC kommer att spela tonen tills du trycker på en annan tangent. När du spelat färdigt, trycker du på 0 eller 9 och pianot stängs av. För att starta pianot igen, skriv bara RUN (och tryck RETURN).

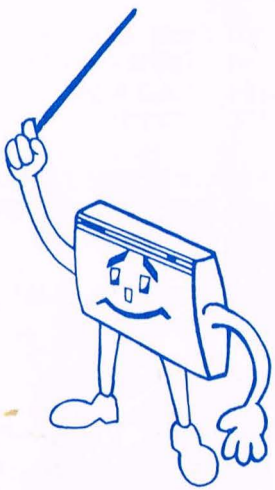
Försök med följande (sjung med om du vill):

```
1  1  5  5  6  6  5
4  4  3  3  2  2  1
5  5  4  4  3  3  2
5  5  4  4  3  3  2
1  1  5  5  6  6  5
4  4  3  3  2  2  1  8
9
```

eller:

```
3  3  4  5  5  4  3  2
1  1  2  3  3  2  2
3  3  4  5  5  4  3  2
1  1  2  3  2  1  1
0
```

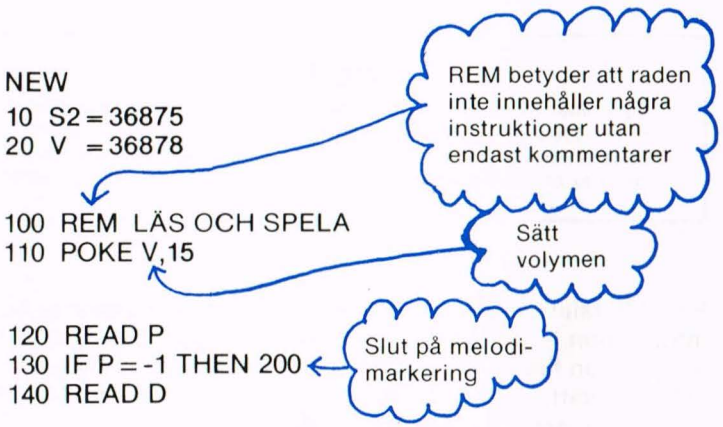
Swing it magistern, swing it!



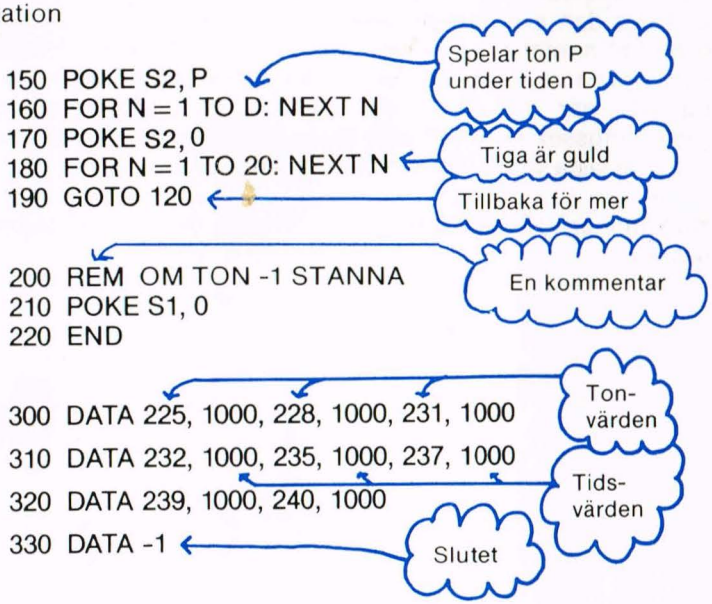
SPELA MELODIER

Genom att använda VIC's stämmor och tabellen för musiktoner, kan du göra dina egna melodier, eller föra över melodier från en sångbok. Följande melodiprogram visar hur man gör:

Skriv:



VIC letar efter information



I den första gruppen instruktioner, lägger du in förkortningar för stämman, vi använder stämma två, och volymen. Nästa grupp instruktioner börjar på rad 100.

Rad 100 innehåller en REM-sats (REM = anmärkning) som förklarar vad sektionen kommer att utföra. Denna sektion innehåller flera loopar eftersom den läser och spelar en ton, och sedan hoppar (loopar) tillbaks till början för att göra om samma sak med en annan ton. Rad 110 skruvar upp volymen.

Nu behöver du bara tala om för VIC vilka toner den skall spela:

VIC TIPS:

Programmen behöver inte starta med rad nummer 1 eller ökas med ett steg. De flesta programmen startar på rad 10 och går i steg om 10. Om du gör på detta sätt kan du gå tillbaks och lägga in extrarader mellan dina tidigare rader. T ex om du vill addera rad 11 och 12 mellan raderna 10 och 20.

Rad 120 talar om för VIC att titta igenom programmet och läsa information — kallad P — om vilken ton den skall spela. Denna information finns i en mystisk rad som vi inte har skrivit ännu. På liknande sätt, talar rad 140 om för VIC att läsa information — kallad D — om hur länge tonen skall ljuda.

Notera speciellt rad 130. Denna rad kommer att stanna programmet när den sista tonen spelats. Utan någon typ av "slut på melodin" nummer, kommer programmet att försöka läsa tonerna som inte har skrivits och det kommer då att bli fel. Rad 130 säger till VIC, när den läser ett värde på —1, att hoppa till en avslutningssektion i rad 200. Du måste komma ihåg att lägga in —1 som sista värde i våra mystiska rader.

Nu kommer VIC att spela en ton, avsluta den och gå tillbaks till nästa:

Rad 150 stoppar (POKE) in tonen som nyss vi läste i rad 120 in till stämma nummer 1, medan rad 160 skapar en tidsfördröjning vars längd beror på värdet som vi läste in i rad 130. På liknande sätt stänger rad 170 och 180 av stämma 1 för en kort period. Rad 190 skickar tillbaks VIC till rad 120 för att läsa nästa ton.

Nu måste vi skriva vår slutsektion. Kom ihåg att vårt "slut på melodi" värde är —1, som sänder programmet till rad 200.

Rad 210 stänger av stämman och rad 220 talar om för VIC att stanna (STOP) körningen av programmet.

Även fast vi skrivit den sista sektionen, så är vi inte riktigt färdiga ännu. Vi måste fortfarande skriva våra mystiska rader för att tala om för programmet vilka toner det skall spela. Dessa mystiska rader är fyllda med DATA. De innehåller information eller data. DATA-kommandot kan placeras var som helst i programmet. När VIC kommer till ett READ-kommando, tittar den efter ett DATA-kommando för att läsa (READ) information.

Denna modul innehåller DATA för C-skalan. Rad 300 innehåller de första tre tonerna. Första numret är noten, 225 — låga C. Det andra numret är tiden, 1000 — ca 1 sekund. Rad 310 innehåller de tre efterföljande tonerna och rad 320 de två sista. Värdena har tagits ur tabellen för musiktoner, tidigare i detta kapitel.

Om raderna ser korrekta ut, skriv RUN. Du kommer att höra en ganska bra C-dur skala! Hör du ett missljud eller två, försök att justera värdet i raderna som börjar i 300.

Ännu en gång: Du kan lägga in DATA-kommandot var du vill i programmet. De kommer att läsas av (READ) en efter en med början i raden det med lägsta numret.

NÅGRA ORD OM POKE

Kommandot POKE tillåter dig att arbeta med VIC på en helt annan nivå. POKE tillåter att du kan ändra innehållet i en speciell minnesposition. Eftersom detta kommando operar direkt på VIC:s minne, är det möjligt att göra misstag genom att POKa in värden i fel position, och fel värden i rätt position! Vi repeterar vad vi sa i kapitel 1: Du kan på inget sätt skada din dator genom att skriva på tangentbordet. Inte ens med POKE. Men du kan orsaka att VIC "dyker" ner någonstans och du tappar kontakten med den. När du är färdig med detta kapitel, försök att skriva:

```
POKE 788,0
```

Du kommer att upptäcka att det enda sättet att återfå kommunikation med din dator är att trycka RUN/STOP och RESTORE. Om kraschen är allvarlig måste du stänga av och sätta på den igen. Detta skadar inte datorn på något vis, men programmet som fanns i den har försvunnit för evigt. Det som händer är att litet tid kommer att gå förlorad. Vi föreslår att du är försiktig med att använda POKE. Det visar också på att det är nödvändigt att ha yttre lagringsminnen typ skivminnen eller kassettstationer som kan lagra program. Likaså skrivare som kan ge en utskrift av ditt program på papper och hjälpa dig med rekonstruktionen av eventuellt försvunna program.

Vi rekommenderar att du gör en kort studie av POKE i bilaga C innan du börjar att använda POKE i din VIC. Var försiktig när du skriver stora värden och dubbelkolla dina program innan du kör dem. En dator som blivit POKad på fel ställe kommer troligtvis att dyka ner någonstans.

I detta kapitel har du lärt dig spela Bach och Beethoven. Du vet hur du kan göra dina grannar galna och längta tillbaks till de gamla goda dagarna när din VIC var tyst, vilket kan skapa nya fina problem med dina vänner. Även om din VIC inte är lika talangfull som Jokkmokks Jocke eller som en Steinway flygel så kan du och din VIC skapa vacker musik tillsammans! Maestro, Please...

6

Prata med din VIC

- Vad heter du?
- Variabler
- Välj en ton
- GET-kommandot

Skriv detta program:

Skriv programmet precis som rutan nedanför visar, och se vad som händer!

```
10 INPUT "GRADER FAHRENHEIT";F
20 PRINT F "GRADER F."
30 PRINT "ÄR"(F-32)*5/9"GRADER C."
40 PRINT
50 GOTO 10
```

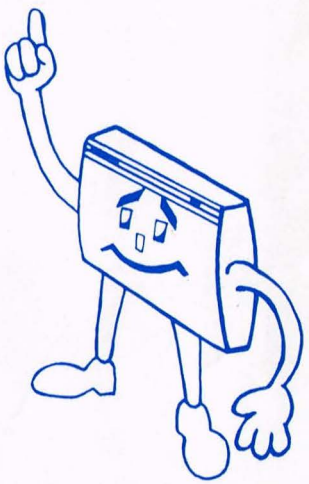
Att skriva instruktionen PRINT ensamt på en rad adderar en blankrad när programmet körs. Försök med och utan denna rad

Skriv RUN och tryck **RETURN**

För att stoppa programmet, tryck **RUN STOP** tangenten.

och tryck

RESTORE



Godmorgon VIC! Ditt uppdrag (och vi vet att du kommer att göra det) är att utföra mirakel när vi trycker på några tangenter!

Bra, din VIC är redo, att svara på dina tangenttryckningar på ett sätt som ingen annan dator har gjort. Sätt dig ner så ska vi ha litet kul!

Håll nere SHIFT tangenten och tryck CLR HOME tangenten.

Skriv på följande tangenter:

N E W och tryck

RETURN tangenten.

Detta är nummer 1 inte L

Tryck på blank-tecken, inte bokstäverna

Glöm inte semikolon

1 I N P U T

" V A D SPACE

H E T E R SPACE

D U ? " ; A $

och RETURN tangenten

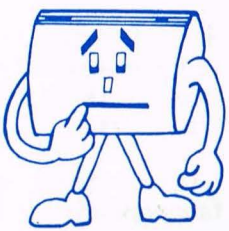
2 P R I N T "

H E J , SPACE " A $

och RETURN tangenten

3 G O T O 2 och tryck på RETURN tangenten.

R U N och tryck på RETURN tangenten.



VIC svarar:

```
1 INPUT "VAD HETER DU";A$
2 PRINT "HEJ,"A$
3 GOTO 2
RUN
VAD HETER DU?
```

Notera att datorn automatiskt adderar fråge-tecknet (?) för att visa att den väntar ett svar.

När markören kommer fram så svarar du med ditt namn. Är ditt namn VIC så skriver du:

V I C och tryck RETURN

Hoppsan! Ditt namn kom farande ut på skärmen! För att sakta ned utskriften trycker du på CTRL tangenten. När du sett ditt namn tillräckligt länge, trycker du på RUN STOP tangenten.

För att läsa in ett annat namn, skriv bara RUN och du kommer att få se samma fråga igen. Frågan är i det här fallet:
VAD HETER DU?

Frågan används för att få in rätt information till VIC. VIC har ett begränsat antal sätt att skaffa sig information från människans värld. Den kanske mest användbara metoden för en dator är att ta in information från ett tangentbord med ett INPUT-kommando. Låt oss gå igenom programmet steg för steg. Här är vad du sagt åt VIC att göra:

STEG 1: Skriv ut meddelandet "VAD HETER DU" på skärmen och vänta sedan på att du skall skriva in tecken från tangentbordet. Ta svaret och kalla det "A$". I vårt exempel blir A$ "VIC". Detta är en typ av förkortning för datorn.

STEG 2: Skriv ut ordet "HEJ" följt av vad du skrev in på tangentbordet. I vårat exempel skrivs HEJ VIC ut.

STEG 3: Gå tillbaks till rad nummer 2.

Rad 2 och 3 tillsammans gör att VIC skriver ut meddelandet HEJ över hela skärmen. Om du ändrar rad 3 till:

3 GOTO 1

så skrivs meddelandet HEJ VIC ut endast en gång och frågan VAD HETER DU kommer upp igen. Med denna ändring ser det ut som VIC börjar bli senil!

När vi kommer till ett INPUT-kommando i programmet, stannar programmet upp och väntar på svar från dig. Det är viktigt att notera att VIC väntar i evighet eller tills du trycker på RETURN-tangenten, vilket som kommer först.

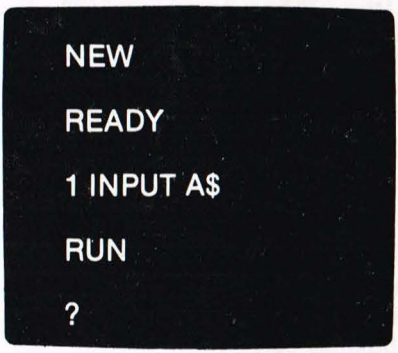
I vårt program ställer vi en vänlig fråga. Om vi inte talar om för VIC vilken fråga den skall ställa vid INPUT-kommandot, skrivs bara ett frågetecken ut. Vi skriver därför istället frågor, för att få en ledtråd om vad vi skall svara med för något.

Tryck RUN STOP och RESTORE samtidigt.

Skriv NEW och

tryck RETURN

1 INPUT A$



När vi skriver RUN
ser skärmen ut så här:

Frågetecknet ensamt skapar mer frågor än det ger svar på. Så låt oss försöka det här:

1 PRINT "VAD HETER DU":INPUT AS

2 PRINT "VAD ÄR DIN FAVORITRÄTT":INPUT BS

3 PRINT

4 PRINT "TACK,"A$"FÖR DITT VÄNLIGA SVAR"

5 PRINT "VI SKALL BJUDA DIG PÅ SPACE "B$" SPACE
FÖR ATT VISA VÅR UPPSKATTNING"

Efter att ha skrivit RUN och tryckt RETURN så ser du att om

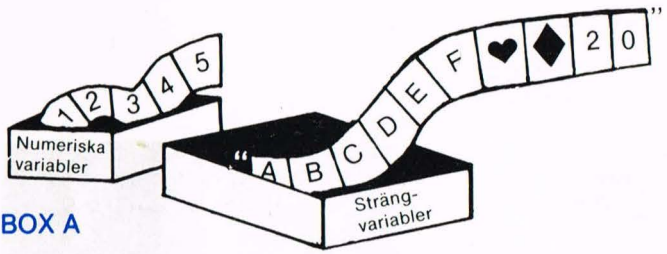
du använder meddelande på rätt sätt, får du ett mycket snyggt resultat.

INTRODUKTION TILL VARIABLER

Många problem i denna bok använder variabler för att förenkla programmet.

Variablerna är mycket värdefulla eftersom de kan användas för att representera siffror, värden, formler, grafik, symboler, ord, fraser t o m hela meningar. Exempel på variabelnamn är X, AB, S2, XS, ABS, S2S. Det enklaste sättet att förklara hur kraftfulla variablerna kan vara är att varje variabel kan representera upp till 255 tecken!

Det finns två typer av variabler: numeriska variabler och strängvariabler. Numeriska variabler används för att lagra nummer (numeriska värden). Strängvariabler för att lagra alla typer av karaktärer och tecken (nummer, siffror, bokstäver, grafik, markörtecken, färgtecken etc).



BOX A

BOX A$

Variabler är som lagerlådor inuti datorn. För att VIC ska veta att lådan endast är för numeriska värden måste du använda ett speciellt namn. Numeriska variabelnamn kan vara ett eller två tecken långa, och det kan vara en bokstav, två bokstäver eller en bokstav och en siffra. Här är några exempel på variabelnamn:

X AB S2 C2 AA ZX

Namnen på strängvariabler kan vara 2 eller 3 tecken långa (inkluderat $-tecknet), och måste alltid innehålla en bokstav från A till Z och ha $-tecknet som sista tecken. Här är några exempel:

X$ AB$ S2$ C2$ AA$ ZX$

Här är ett kort program som visar hur du använder variabler:

```
10 A$ = "VIC 20"

20 PRINT"HEJSAN,"AS

RUN
```

VIC kommer att skriva ut HEJSAN VIC 20. Varför? Därför i rad 10 talar vi om för VIC att variabeln A$ har innehållet "VIC 20". Försök detta:

```
10A = 2

20B = 3

30C = 4

40PRINT A*B*C
```

I detta exempel är A*B*C lika med 2*3*4 därför att vi lagrade in värderna 2, 3 och 4 i variablerna A, B och C.

Här kommer det sista exemplet som använder två typer av variabler i INPUT-kommandot. INPUT-kommandot gör att personen som kör programmet kan bestämma vad variablerna skall innehålla.

```
10PRINT"VILKEN TEXT VILL DU ATT X$ SKALL STÅ FÖR":
  INPUT X$

20PRINT"VILKET NUMMER VILLD U ATT X SKALL STÅ FÖR":
  INPUT X

30PRINT"NU STÅR X$ FÖR" X$

40PRINT"OCH X STÅR FÖR" X

RUN
```

VÄLJ EN TON

Detta exempel spelar den ton du väljer.

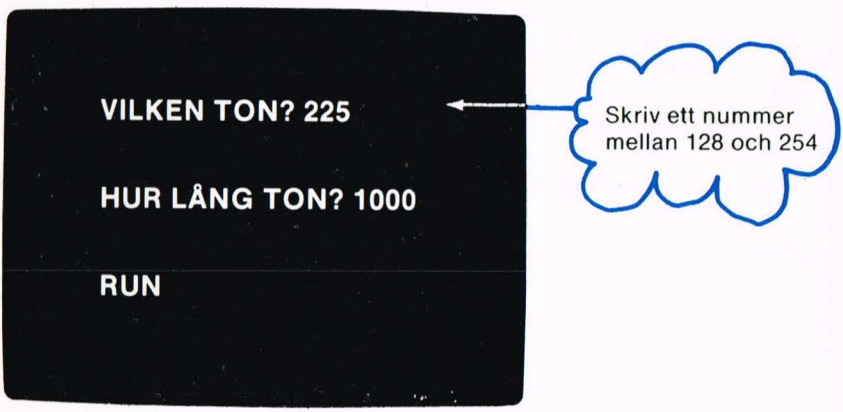
Skriv:

```
NEW
10 INPUT "VILKEN TON";H
20 IF H=0 THEN 90
30 INPUT "HUR LÅNG TON";L
40 POKE 36878,15
50 POKE 36875,H
60 FOR T=1 TO L:NEXT
70 POKE 36878,0
80 GOTO 10
90 END
```

Denna bokstav är en numerisk variabel och används för att definiera eller mata in ett värde

Om en ton fortsätter att spela tryck RUN STOP RESTORE

Efter att du skrivit RUN och tryckt RETURN bör skärmen se ut så här:



Skriv in värdena och tryck RETURN och lyssna på tonen och hur länge den ljuder. Efter varje ton kan du skapa en ny, och få en uppfattning om hur dessa mysteriska POKE-kommandon förvandlar värdena till ljud.

De två första raderna (10 och 20) frågar dig vilka värden tonen skall ha. Dessa värden återfinner vi i rad 50 och 60. Rad 30 är dörren ut ur programmet när du experimenterat färdigt (skriv in värdet 0 på frågan ("VILKEN TON"). Rad 70 stänger av tonen som blev aktiverad i rad 40 och 50.

Om du vill veta alla detaljer om musik och toner i VIC, läs kapitel 5.

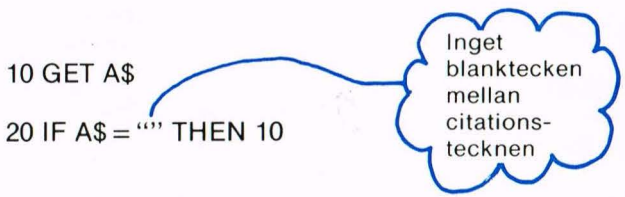
GET KOMMANDOT

Nu när du lärt dig INPUT-kommandot, skall vi fortsätta med ett mera användbart kommando för att få information från tangentbordet.

GET-kommandot används för att läsa ett tecken i taget från tangentbordet. Personen framför tangentbordet behöver inte trycka på RETURN-tangenten. Så här ser kommandot ut:

```
10 GET A$
```

Hur gör vi för att få programmet att stanna upp och vänta tills någon har tryckt på en tangent? Vi lägger programmet i en loop med ett IF . . . THEN-kommando som kontrollerar svaret.



Vad är meningen med detta? Dessa två enkla rader tillåter att ditt program gör en paus tills personen vid tangentbordet trycker på en tangent. Detta är bra när man vill frysa en skärm för att kunna läsa färdigt den och sedan gå vidare.

Här har vi utökat exemplet litet:

```
10 GET A$

20 IF A$ = "" THEN 10

30 IF A$ = "A" THEN PRINT "KYCKLINGSOPPA"

40 IF A$ = "B" THEN PRINT "SPAGETTI"
```

```
50 IF A$ = "C" THEN PRINT "STEK"

60 GOTO 10

RUN
```

När du skrivit in programmet och skrivit RUN kommer programmet att vänta tills du trycker på en tangent. Om tangenten som du tryckte var ett "A", skrivs ordet "KYCKLINGSOPPA" ut på skärmen. Var tangenten ett "B" skrivs ordet "SPAGETTI" ut och var tangenten ett "C" skrivs ordet "STEK" ut på skärmen. Du har nu fått VIC att skriva ett helt ord men endast en tangenttryckning!

Nu skall vi göra det längsta programexemplet hittills. Ett program som visar en praktisk tillämpning av GET och PRINT-kommandon för att skapa en datoriserad kokbok! Bli inte rädd för storleken på detta program. Programmet består mest av enkla PRINT-kommandon. Notera att A$ är den enda variabel som används i programmet och att GET-kommandot används för att stoppa upp programmet.

```
10 PRINT " SHIFT CLR HOME VÄLJ EN RÄTT"
20 PRINT "FRÅN MENY:"
30 PRINT
40 PRINT "A...KYCKLINGSOPPA"
50 PRINT "B...SPAGETTI"
60 PRINT "C...STEK"
200 GET A$:IFA$ = ""THEN 200
210 IF A$ = "A" THEN 500
220 IF A$ = "B" THEN 700
230 IF A$ = "C" THEN 900
490 GOTO 200
500 PRINT " SHIFT CLR HOME MICKES KYCKLINGSOPPA"
510 PRINT
520 PRINT "TAG 1 STYCKEN KYCKLING"
```

Försök att skriva dina egna recept! Du behöver bara ändra rubrikerna och receptinnehållet...

Detta betyder att trycker du någonting annat händer ingenting och programmet fortsätter att vänta på att du skall trycka A, B eller C.

```
530 PRINT "AVLIVA DEN OCH PLOCKA";
540 PRINT "BORT FJÄDRAR OCH FÖTTER.";
550 PRINT "KOKA UPP 10 LITER VATTEN";
560 PRINT "LÄGG I KYCKLINGEN OCH";
570 PRINT "LÅT KOKA I 2 TIMMAR ELLER";
580 PRINT "TILLS DET LUKTAR GOTT."
590 PRINT
600 PRINT "TRYCK PÅ EN TANGENT
  FÖR ATT FORTSÄTTA"
610 GET A$:IF A$ = "" THEN 610
620 GOTO 10
```

Notera att delar av meningarna skrivs på separata rader för att göra det lättare att läsa.

Skriva enbart PRINT gör att en blankrad skrivs ut.

```
700 PRINT "  SHIFT  CLR HOME  "MAMMAS SPAGETTI"
710 PRINT
720 PRINT "BRYN 4 HG KÖTTFÄRS"
730 PRINT "MED 1 LÖK OCH"
740 PRINT "GRÖN PEPPAR. ADDERA"
750 PRINT "2 HG TOMATPURE,"
760 PRINT "TVÅ SKEDAR TOMATSKAL"
770 PRINT "2 DL VATTEN"
780 PRINT "3 KLYFTOR VITLÖK, SALT"
790 PRINT "OCH PEPPAR, OREGANO"
800 PRINT "LÅT SJUDA EN TIMMA"
810 PRINT "SERVERA MED KOKTA"
820 PRINT "MAKARONER"
830 GOTO 590
```

```
900 PRINT " [SHIFT] [CLR HOME]      STEK "
910 PRINT
920 PRINT "TAG EN KOKT STEK"
930 PRINT "OCH ETT STEKT ÄGG"
940 PRINT "SERVERA TILLSAMMANS"
950 PRINT "MED GRÖNSAKER"
960 GOTO 590
```

Om du skrivit programmet riktigt och skrivit RUN, kommer skärmen att se ut så här:

Nu väntar VIC på att du ska trycka en tangent. Om du trycker på en annan tangent än A, B eller C händer ingenting (rad 200 till 490 gör detta). Om du trycker A så får du receptet för Lasses kycklingsoppa. Ganska förskräcklig eller hur? Det syns att Lasse är ungkarl.

Programmet kan utvidgas mycket enkelt av dig själv. För att addera rätter till menyn, skriv till några PRINT-kommandon efter rad 60 och några IF . . . THEN-kommando efter rad 230, och skriv in receptet där det finns plats. I din sista rad ska det stå GOTO 590, där programmet talar om för personen att trycka på en tangent för att fortsätta. Detta kommer att hålla kvar receptet i skärmen tills du är färdig med det.

Du kan givetvis använda exemplet till annat än recept. Varför inte namn och adress? Istället för meny används första bokstaven i efternamnet. Som recept skrivs personens namn, adress och telefonnummer ut på skärmen. Du kan säkert komma på andra program där GET och INPUT kan användas. Detta är verklig datormakt — att kunna säga till datorn att göra vad DU vill.

7

Introduktion till programmering

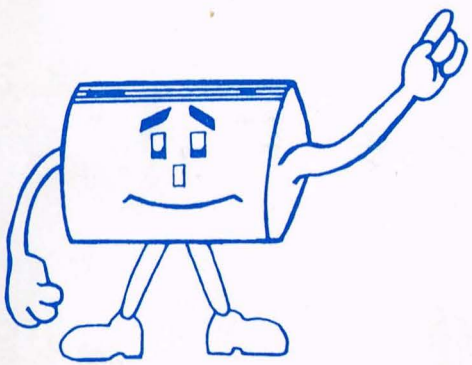
- Ditt första BASIC-kommando
- Slumptal

Skriv detta program:

Skriv programmet precis som rutan nedanför visar, och se vad som händer!

```
10 PRINT " SHIFT CLR HOME "
20 PRINT CHR$(205.5 + RND(1));
30 GOTO 20
```

För att stoppa programmet tryck RUN STOP tangenten.



DINA FÖRSTA BASIC PROGRAM OCH HUR DE FUNGERAR

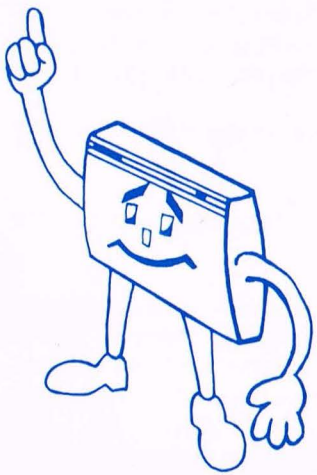
Ända tills nu har du varit tålmodig och skrivit in alla programexempel, kanske utan att förstått hur de fungerat. Detta kapitel kommer att förklara vad de små programexemplen gjorde för något, och visa dig litet om hur du programmerar din VIC.

PROGRAM 1: Ditt namn på skärmen
Kapitel 2

```
10 PRINT " SHIFT CLR HOME

20 FOR T = 1 TO 300: NEXT

30 PRINT "ditt namn"

40 FOR T = 1 TO 300: NEXT

50 GOTO 10
```

Detta program och alla som följer är tagna från exempelprogrammen i början på varje kapitel.

Det ser inte ut att vara mycket till program, men när du förstått vad som sker i programmet vet du hur animering går till. PRINT-kommandot i rad 10 innehåller tecknet "rensa skärmen" mellan citationstecknen. Om du försöker skriva raden utan citationstecken, försvinner din programrad från skärmen innan du hann skriva färdigt den. VIC läser endast in den raden när du trycker RETURN och markören befinner sig på raden.



VIC TIPS: CITATIONSTECKEN

Låt oss tala om vad som händer när du trycker på tangenten med citationstecken. Första gången du trycker på citationstecknet, händer någonting konstigt. Om du trycker på HOME, CLR, markör upp, ner, vänster eller höger får du ett reverserat grafiskt tecken. Detta händer också om du håller CTRL-tangenten nedtryckt och trycker på någon av VIC's färgtangenter. Citationstecken används i datorprogram och VIC använder citationstecknet som ett programmeringskommando, därför skriver VIC ut specialtecken för att visa att du tryckt på en färg eller markör-tangent. Till exempel när du skriver " och SHIFT får du ett reverserat hjärta på skärmen. Om du ser detta tecken i ett program så vet du att det betyder rensa skärmen. Andra symboler står för andra kontrolltangenter. När du tryckt på CLR HOME citationstangenten en andra gång så fungerar markör och färgtangenterna normalt igen.

INST-tangenten kan orsaka liknande effekt. När denna tangent trycks ned kommer varje blanktecken som skapades på raden att uppträda som om du skrivit ett citationstecken före blanktecknen. Alla färg och markörtangenter kommer att skrivas ut som om de vore mellan citationstecken. Dessutom kommer DEL-tangenten att generera ett speciellt tecken i dessa blanktecken. Detta får effekten att tecken raderas när programmet listas eller utskrifter görs vid programkörning.

Rad 20 i programmet kalla för en tidsfördröjningsloop. Denna rad innehåller två stycken BASIC-kommandon vilka separeras med ett semikolon. Vad som händer är att VIC räknar från ett till hundra utan att göra någonting annat. Detta gör att programmet blir långsammare. (Försök att radera rad 20 och 40 och kör programmet. Det blinkar för fort.)

I rad 30, skrev du in ditt namn mellan citationstecknen (vet du inte vad du heter, skriv då namnet på din bror). Denna rad åstadkommer att ditt (eller din brors) namn skrivs ut på skärmen. Det finns ingenting annat på skärmen eftersom rad 10 har rensat den.

Rad 40 är ännu en tidsfördröjning för att ge dig tid att läsa ditt namn på skärmen.

Rad 50 säger till VIC att gå till "go to" rad 10 för att göra om samma saker igen.

Efter att ditt namn skrivits ut på skärmen, är det en tidsfördröjning på 1 sekund, varefter skärmen rensas. Efter ännu 1 sekund, dyker namnet upp på skärmen igen i samma position. Sedan raderas den igen osv. Eftersom bokstäverna skrivs ut i samma positioner i skärmen ser det ut som de blinkar.

Experimentera med detta program. Du kan göra tidsfördröjningarna längre genom att ändra värdet 300 i rad 20 och 40.

PROGRAM 2: Många hjärtan
Kapitel 3

```
10 FOR H = 1 TO 505
20 PRINT " ♥ ";
30 NEXT
40 FOR C = 8 TO 255 STEP 17
50 POKE 36879, C
60 FOR T = 1 TO 500: NEXT
70 NEXT
80 GOTO 40
```

Detta program fyller skärmen med färgglada hjärtan. Programmet introducerar formatteringstecken i PRINT-kommandot och POKE-kommandot för att ändra bakgrunds- och ramfärgerna.

Rad 10 lägger upp en loop som räknar från 1 till 505. Vi vill ha ut 505 hjärtan på skärmen, därför att det finns 506 positioner på skärmen. Om vi skrev ut 506 tecken, så skulle skärmen rulla upp en rad (scrolling) och det skulle bli mindre hjärtan på skärmen.

Rad 30 avslutar loopen. Så länge värdet H är 505 eller mindre, skriver programmet ut hjärtan på skärmen. När programmet skrivit ut 505 hjärtan fortsätter programmet med raden efter (rad 40).

Rad 40 sätter upp en loop. Rad 50 ändrar bakgrunds- och ramfärgerna. Variabeln C definieras som en serie av värden från 8 till 255, ökande med steg (STEP) om 17. Varje gång programmet kommer till rad 70 (NEXT), adderas 17 till det tidigare värdet C och summan läggs in i C. Detta gör att en serie med färgkombinationer kommer fram, svart bakgrund och svart ram, vit bakgrund och vit ram, osv för alla 8 färgerna. När variabeln C har gått igenom alla dessa färger kommer den till 7 olika färgkombinationer. (Se bilaga I för färgkoder).

I rad 50 finns kommandot som ändrar färgen. Värdet som finns i variabeln C lagras i minnespositionen 36879. Denna position tillhör VIC-kretsen, och inte det normala minnesområdet.

Rad 60 är en tidsfördröjningsloop. Om denna rad tas bort, ändras färgerna så fort att du får ont i huvudet.

Rad 70 avslutar loopen som började i rad 40. Notera att det lika gärna kunde stå 70 NEXT C, men det är inte nödvändigt.

Rad 80 sänder tillbaka programmet så att det börjar skifta färger igen. Programmet kommer att gå i evighet om du inte trycker på STOP-tangenten eller stänger av VIC. Skriv POKE 36879,27 efter att du tryckt STOP och skärmen intar de normala färgerna.

PROGRAM 3: Träna med TV
Kapitel 4

```
10 PRINT " CLR HOME ";
20 PRINT " ◪ ○ ◪ "
30 PRINT " □ ▦ □ "
40 PRINT " ◪ □ ◪ "
50 FOR T = 1 TO 300: NEXT
60 PRINT " CLR HOME ";
70 PRINT " □ ○ □ "
80 PRINT " ◪ ▦ ◪ "
90 PRINT " □ ▌ ▌ "
100 FOR T = 1 TO 300: NEXT
110 GOTO 10
```

Detta program påminner om det första programmet vi gjorde, det som skrev ut ditt namn på skärmen. Detta program skriver ut en figur, gör en paus och skriver sedan ut en liknande figur i samma position. Huvudet och kroppen på VIC-personen stannar i samma position medan armarna och benen förflyttar sig. Detta ger illusionen av att figuren rör på sig.

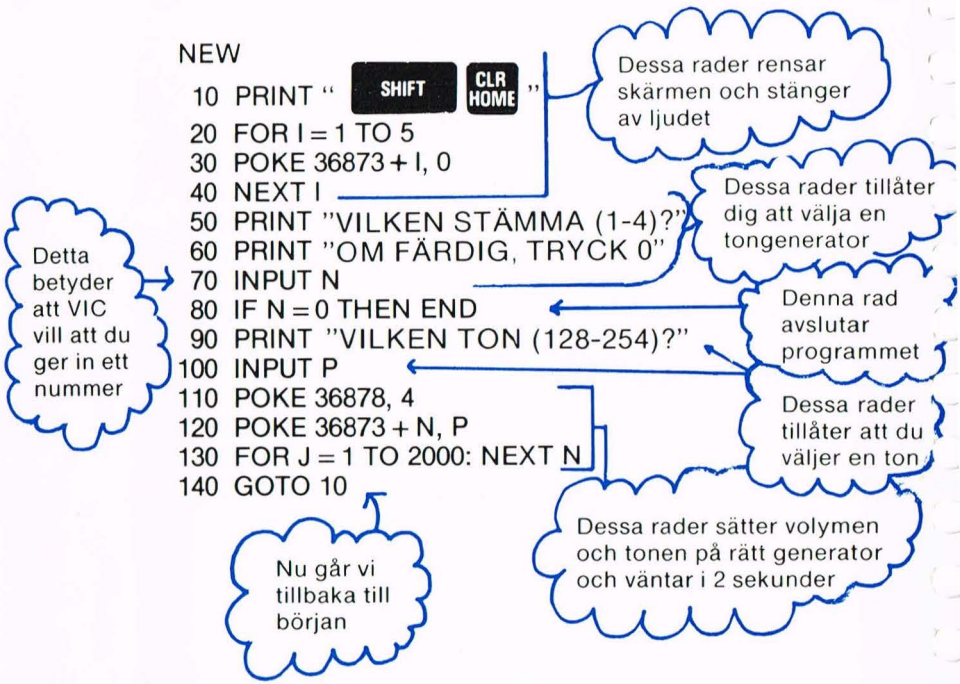
Rad 10 och 60 förflyttar markören upp till det övre vänstra hörnet vilket vi kallar för "home" positionen. Detta tvingar bilderna av VIC-personen att skrivas ut på samma ställe i skärmen varje gång.

Rad 20, 30 och 40 ritar första bilden av VIC-personen.

Rad 50 och 100 är en tidsfördröjningsloop, för att figuren skall synas tillräckligt länge på skärmen.

Rad 70, 80 och 90 ritar ut den andra bilden av figuren på skärmen. Denna utskrift går så fort att du inte hinner se den — ditt öga uppfattar förändringen som ögonblicklig.

PROGRAM 4: Välj en ton
Kapitel 5



När du kontrollerar att alla rader är riktigt inskrivna kan du köra programmet (skriv RUN och tryck RETURN). Programmet tillåter att du väljer en stämma och en ton som hörs i ca 2 sekunder. Tonen stängs av och programmet frågar efter en ny stämma och ton. Detta program är bra om du vill lära dig VIC's musikmöjligheter, så försök alla kombinatoner. När du vill sluta programmet, svarar du 0 som stämma. Du kanske tror att det är någonting fel när du väljer ton 254 i stämma tre och inte hör någonting. Det är faktiskt inte något fel, denna ton är för hög för att människans öra skall uppfatta det. (Försök testa tonen på din hund istället!)

PROGRAM 5: Temperaturberäkning
Kapitel 6

```
10 INPUT "GRADER FAHRENHEIT";F
20 PRINT F "GRADER F."
30 PRINT "ÄR" (F-32)*5/9 "GRADER C."
40 PRINT
50 GOTO 10
```

Detta program lär dig att använda INPUT-kommandot. Detta kommando tillåter programmet att stanna och fråga efter information från operatören (personen som kör programmet).

Rad 10 gör att meddelandet "GRADER CELSIUS?" kommer upp på skärmen. Orden mellan citationstecknen i ett INPUT-kommando behandlas på samma sätt som i ett PRINT-kommando. Den enda skillnaden är att det sista tecknet följs av ett frågetecken och programmet stannar för att vänta på information.

Rad 20 skriver ut värdet C, som är precis vad du nyss skrev in. Rad 30 skriver ut resultatet av beräkningen. Det ensamma PRINT-kommandot i rad 40, gör att en blankrad skrivs ut.

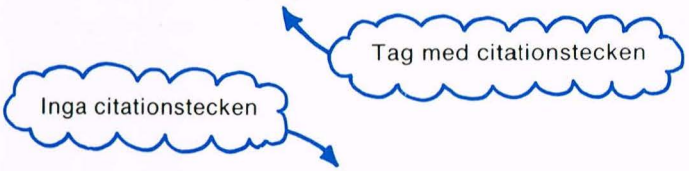
Den sista raden, rad 50, gör att programmet hoppar tillbaka till början för att fråga om mer information. Om första frågan ställs igen, och du kan få fler temperaturer beräknade. Om du är färdig, håll nere STOP-tangenten och tryck på RESTORE-tangenten. Det finns inget annat sätt att tala om för programmet att du är färdig.

PROGRAM 6: Slumptal
Kapitel 7

```
10 PRINT " SHIFT CLR HOME "

20 PRINT CHR$(205.5 + RND(1));

30 GOTO 20
```

Ett litet nätt program som ger slumpmönster över hela skärmen. Som du kanske förstår är det rad 20 som ligger bakom detta.

CHRS-funktionen ger ett tecken baserat på ett värde från 0 till 255. Varje tecken som VIC skriver ut på skärmen kodas på detta sätt (se bilaga H). För att ta reda på koden för något tecken, skriv bara. PRINT ASC("X"), där x är tecknet som du vill kontrollera.



Skriv sedan PRINT CHRS(X), där x är numret som VIC gav dig. Du ser, det fungerar!

Försök att skriva PRINT CHRS(205);CHRS(206). Detta skriver ut de två högra grafiktecknen på tangenterna M och N. Detta är de två tecken som används för att göra mönstret.

Genom att addera formeln 205.5 + RND(1), tar VIC ut ett slumptal mellan 205.5 och 206.5. Chansen är fifty fifty att talet skall bli över 206. När CHRS-funktionen arbetar, så struntar den i bråkdelen av talet. Därför kommer hälften av bokstäverna att skriva ut med koden 205 och den andra hälften med 206.

Om du vill experimentera med programmet kan du ändra talet 205.5 ungefär 10 enheter uppåt eller nedåt. Med detta blir chansen större att du får ett tecken som syns på skärmen.

SLUMPTAL (RND)

Slumptalsfunktionen är den mest användbara och det roligaste kommandot i BASIC när man skall göra olika sorters spelprogram.

Raden X = RND(1) gör att VIC väljer ut ett slumptal mellan 0 och 1, och placerat det i variabeln X. Detta ger en rad möjliga värden för X:

0<X<1

När vi arbetar med slumptal så vill du kanske ha talet inom ett speciellt intervall. Till exempel du kanske vill ha ett värde mellan 0 och 3. Då multiplicerar du X med 3. Det nya intervallet är:

0<X<3

Om du vill ha ett tal mellan 10 och 20, hur gör du då beräkningen? Först så multiplicerar du slumptalet med 10 och får intervallet:

10<X<11

Sedan adderar du 10 till ditt resultat och du får talet inom intervallet:

10<X<20

Så formeln för ett slumptal mellan 10 och 20 är:

X = RND(1)*10 + 10

Hittills har vi lärt oss att ändra intervallet för slumptalet. Men resultatet innehåller en del onödiga decimaler som vi inte vill ha när vi till exempel skall kasta tärning eller få ett nummer mellan 1 och 10. Funktionen för att rensa bort decimalerna heter INT. Denna kommer att kapa bort alla decimaler från talet. Formeln för ett slumpmässigt heltal från 10 till 20 med INT-funktionen tillagd ser ut så här: X = INT(RND(1)*10+20)

Området för möjliga resultat är nu:

10<=X<=19

Men vänta! Den övre gränsen har ändrats från 20 till 19. Varför? Därför att intervallet var förut alltid mindre än 20. INT-funktionen kommer att ta bort alla decimaler från ett tal som är större än 19 och mindre än 20 och detta resulterar i 19. I andra änden av

intervallet, blir resultatet mellan 10 och 11 nerkapat till 10 jämt. Om vi fortfarande behöver tal inom intervallet 10 till 20, så bör formeln se ut så här:

X = INT(RND(1)*11 + 10)

Slumptalet multipliceras för att expandera intervallet och adderas för att flytta intervallet.

Den generella formeln för att få ett slumptal inom ett visst intervall är:

X = INT(RND(1)*a) + b

Där a representerar antalet möjliga resultat och b är det lägsta numret i intervallet.

BILAGOR

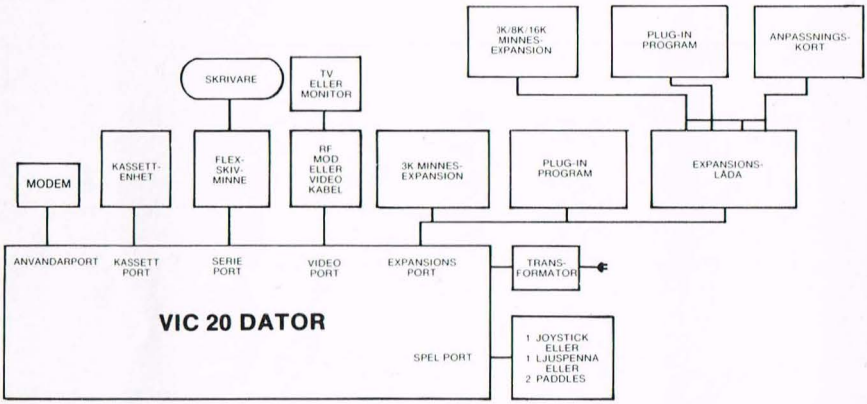
BILAGA A: VIC TILLBEHÖR EN INTRODUKTION

Detta är en handbok för nybörjare så vi ska inte gå igenom alla yttre enheter som kan anslutas till din VIC.

Varje enhet har sin egen handbok som talar om hur den ansluts och används. Den ger också roliga och intressanta tips.

Commodore har konstruerat VIC så att den kan växa efter dina behov. Schemat nedanför visar vilka enheter som kan anslutas till VIC.

VIC 20 DATORSYSTEM



SPECIELLA PLUG-IN PROGRAM

Super expander kort — 3K extra minne (konverterar din VIC till 8K)
Högupplösande grafik och plottnings-kommando
Definierade funktionstangenter

Programmerarkort — Programmerarens verktygslåda
Maskinspråkmonitor
Definierade funktionstangenter (förprogrammerade)
Användardefinierade funktionstangenter

KASSETTENHET

Den första enhet som du troligtvis kommer att skaffa är en kassettenhet. På ett kassettband kan flera tusen tecken lagras (bokstäver och siffror). Stora program kan flyttas in eller ut från VIC's minne relativt snabbt utan att du behöver skriva in programmen varje gång. (Se bilaga B — Att arbeta med kassettenheten.)

Kassettenheten ansluter du på VIC's baksida. Det finns en speciell kontakt för detta. Det finns en mängd inspelade band med olika program som du kan köpa och använda till din VIC.

EXPANSIONSLÅDA

Expansionslådan ansluter du i VIC's expansionsport. Denna låda ger dig möjlighet att ansluta flera kort samtidigt. Det finns plats för 6 stycken kort, programkort, minneskort eller anpassningskort (interface). Exempel på anpassningskort är IEEE-488 vilket ger dig möjlighet att använda PET/CBM's rika urval av enheter.

FLEXSKIVMINNE

VIC's flexskivminne har en station och kan lagra upp till 165.000 tecken (bokstäver och siffror). Stora program kan flyttas mellan flexskivan och VIC på bråkdelar av sekunder. Inga långa väntetider för att hitta rätt program på kassetten. Denna enhet ansluter du på VIC's seriekontakt.

IEEE-488 ANPASSNINGSKORT

IEEE-488 är en universell standard som ger dig möjlighet att använda PET/CBM-enheter. Det finns en mängd av olika skrivare, flexskivminnen och dessutom hundratals olika laboratorieinstrument.

SKRIVARE

Till VIC finns en skrivare som ansluts till VIC's serieport. På skrivaren kan du göa utskrifter av olika slag, till exempel programlistningar, tabeller osv.

SPELPORT

VIC har en spelport där du kan ansluta joysticks, ljuspenna eller paddles. Med dessa kan du göra roliga och intressanta spel utan att behöva använda tangentbordet.

TELEKOMMUNIKATION

Du har kanske hört talas om datorer som kan kommunicera med andra datorer och få information om ..... VIC har redan denna möjlighet inbyggd.

VIC's användarport innehåller en RS232 anpassning för att utbyta information via ett billigt telefonmodem.

MJUKVARA

Förutom all denna trevliga hårdvara finns det en mängd med intressanta mjukvara.

En del av denna mjukvara kan anslutas som ett programkort på VIC's baksida. När du slår på din VIC är mjukvara färdig att köras. En del finns på kassett eller flexskiva och många kan du hitta i den serie med läroböcker och kort som finns för att lära dig allt om datorer.

Detta har varit en kort inledning till alla tillbehör som finns till VIC.

FRÅGA DIN NÄRMASTE ÅTERFÖRSÄLJARE OM DE SENASTE TILLBEHÖREN

BILAGA B:
ATT ARBETA MED KASSETTENHETEN

Kassettenheten är den första enhet som de flesta VIC-ägare köper. Denna enhet gör det möjligt att spara program för att senare läsa tillbaka dem. Du kan också spara data (information) på kassettbandet. Detta kapitel kommer att hjälpa dig när du har en kassettenhet.

Det första kommandot som du använder tillsammans med kassettenheten är SAVE-kommandot (spara). När du ger detta kommando till VIC så svarar den med "PRESS PLAY & RECORD ON TAPE" (tryck på PLAY och RECORD på kassettenhet). Du måste hålla RECORD-tangenten nedtryckt och sedan trycka på PLAY-tangenten. Båda tangenterna skall vara nere när VIC sparar programmet. (Om inte RECORD-tangenten går att trycka ned, titta på baksidan av kassetten. Om det finns ett fyrkantigt hål går det inte att spela in på bandet. Bandet är så kallat skrivskyddat. Detta kan du även göra på ett vanligt band genom att peta bort fliken som täcker hålet.) När PLAY-tangenten trycks ned visas meddelandet "OK" och ordet "SAVING" kommer upp på skärmen. Efter ett tag, beroende på hur långt programmet är, skrivs meddelandet "OK" och "READY" ut på skärmen och SAVE-kommandot har utförts. Du kan ge programmet ett namn på bandet genom att skriva in en text mellan citationstecknen efter SAVE (se bilaga C).

När du sparat programmet är det bäst att du kontrollerar att det verkligen blivit inskrivit på bandet helt riktigt. Det är möjligt att bandet till exempel var skadat. Därför finns ett kommando som heter VERIFY. Kommandot kontrollerar om innehållet på bandet stämmer överens med det i VIC's minne.

När du skriver in kommandot VERIFY svarar VIC med "PRESS PLAY ON TAPE" (tryck PLAY på kassettenheten). Om du lägger till ett namn efter kommandot så letar VIC efter program med det namnet på bandet. Om inget namn anges, jämförs det första programmet med det i VIC's minne. När du trycker ned PLAY-tangenten svarar VIC med "OK" och "SEARCHING" (OK, söker). Sedan listar VIC ut alla programnamn utom det den söker eftr. När den hittat programmet skriver den ut "FOUND" och "VERIFYING". VIC jämför nu programmet på bandet med det program som finns i minnet. Hela programmet kontrolleras även om det finns ett fel direkt i början. När programmet är slut och inget fel upptäckts skrivs "OK" och "READY" ut på skärmen. Om ett fel upptäcks svarar VIC med "VERIFY ERROR". Om du får ett fel försök att använda nya kassettband.

För att läsa tillbaka ett program från kassetten till VIC så använder du kommandot LOAD. VIC säger då till dig att trycka på PLAY-tangenten (PRESS PLAY ON TAPE), och programmet kommer att laddas in från kassetten. Om programladdningen lyckades skrivs meddelandet "READY" ut. Är det svårigheter att läsa kassettbandet svarar VIC med "LOAD ERROR". Om bandet är mycket dåligt kommer kanske konstiga saker att hända med din VIC. Konstiga tecken kommer upp på skärmen, allting kan hända. Om detta händer är det bäst att stänga av VIC och sätta på den igen.

Nu skall vi behandla ett mera komplex sätt att arbeta med kassettstationen, nämligen att spara och hämta data från kassettband.

Det första du måste göra för att kunna skriva in data på kassettbandet är att öppna en fil med kommandot OPEN som följs av tre tal och ett namn. Så här:

OPEN 1,1,1,"HEJSAN"

Det första numret är filnumret som du använder senare i programmet för att referera till. Det andra är enhetsnumret, vilket är 1 för kassettenheten. Det tredje numret är 0 för läsning, 1 för skrivning och 2 för skrivning med markeringen "end of tape" (slut på bandet) inskriven efter filen. Namnet "HEJSAN" läggs in på bandet som identifiering av filen. Detta gör att flera filer kan skrivas in på samma band. Notera att VIC INTE söker rätt på ett ledigt utrymme på bandet och när den läser så letar den efter en fil med det namn som angavs. OPEN-kommandot ovan skapar en fil på bandet, kallad "HEJSAN", för inskrivning av data. Senare i programmet refererar du till filen som fil #1.

För att skriva in data på filen används kommandot PRINT#. #-tecknet följs av filnumret (i ovanstående exempel en 1), sedan ett kommatecken följt av de objekt (texter, variabler, siffror) som skall skrivas ut. Detta kommer att skrivas ut på filen i samma format som det skrivs ut på skärmen. Ett kommatecken gör att några extra blanktecken skrivs ut, vilket är onödigt att få på bandet eftersom de tar lika mycket plats som ett vanligt tecken. Normalt används ett semikolon för att separera objekten.

Du kommer att upptäcka att kassettenheten inte går hela tiden utan startar och stannar ibland. Detta beror på att VIC buffrar data först. Med detta menas att VIC inte skriver ut datat på en gång utan mellanlagrar det i ett speciellt buffertminne i VIC. I bufferten får det plats 192 tecken. När VIC lagt in det 192:a tecknet i bufferten, töms den genom att innehållet skrivs ut på bandet.

När du skrivit in alla data på bandet så måste filen stängas med CLOSE-kommandot. Kommandot följs av det filnummer som du vill stänga. Detta är nödvändigt på grund av bufferten. De tecken som finns kvar i bufferten måste skrivas ut till bandet. Att tömma bufferten klarar CLOSE-kommandot av, dessutom skrivs en markering (end of file) in på bandet som talar om att filen är slut. (Om det tredje talet är 2 skriv en "end of tape" markering in på bandet.

För att läsa tillbaka informatinen från bandet så måste filen först öppnas för läsning (tredje talet = 0). VIC kommer att söka efter filnamnet, om något angavs, och stanna när den hittat filen. Nu kan du använda INPUT# och GET#-kommandona för att läsa in data från bandet. Om bufferten är tom när INPUT# eller GET#-kommandot utförs, fylls den med data från kassettbandet. Data läses från bufferten och inte direkt från bandet.

Skillnaden mellan INPUT# och GET# är att INPUT# läser in en hel variabel från bandet, antingen en sträng eller numerisk variabel. GET# läser bara in ett tecken i taget. Om du experimenterar litet med GET#-

kommandot är upptäcker du att det sista tecknet i varje objekt är CHR$(13), vilket står för RETURN. Detta tecken används för att separera data på bandet så att INPUT#-kommandot vet när variabeln slutar.

Här är tre programexempel som visar hur du kan experimentera med datafiler på kassettband. Det andra programmet skall köras efter du kört det första och spolat tillbaka bandet. Programmet visar hur du kan få tillbaka data genom att använda kommandot INPUT#. Det tredje programmet använder GET#-kommandot för att läsa tillbaka data och skriver inte bara ut tecknet utan också kodvärdet.

PROGRAM 1: SKRIV TILL BAND

```
10 PRINT" SHIFT CLR HOME SKRIV TILL BANDPROGRAM"
20 OPEN 1,1,1,"DATA FILE"
30 PRINT "SKRIV IN DATA SOM"
40 PRINT "SKALL LAGRAS PÅ BANDET CTRL RVS ON STOP CTRL ø RVS OFF
50 PRINT
60 INPUT"DATA";A$
70 PRINT#1,A$
80 IFA$ <> "STOPP"THEN50
90 PRINT
100 PRINT "STÄNGER FILEN"
110 CLOSE 1
```

PROGRAM 2: LÄSA BAND MED INPUT#

```
10 PRINT" SHIFT CLR HOME LÄSA BANDPROGRAM
20 OPEN 1,1,0,"DATA FILE"
30 PRINT "FILEN ÖPPNAS"
40 PRINT
50 INPUT#1,A$
60 PRINT A$
70 IF A$ = "STOPP"THEN END
80 GOTO 40
```

PROGRAM 3: LÄSA BAND MED GET#

Rad 10 till 40 är samma som i PROGRAM 2

```
50 GET#1, A$
60 IF A$ = ""THEN END
70 PRINT A$,ASC(A$)
80 GOTO 50
```

BILAGA C: VIC BASIC

Denna handbok har hittills introducerat dig för ett fåtal kommandon i språket BASIC. Du vet litet grann om programmering och hur program ser ut. I denna bilaga finns en komplett lista på alla regler (SYNTAX) som gäller för de olika BASIC-kommandona, plus en beskrivning av vad de utför. Du rekommenderas att experimentera med alla kommandon för att komma ihåg att du inte kan skada din VIC genom att skriva program. Det bästa sättet att lära sig är att undersöka saker.

Denna bilaga har delats upp i fyra olika delar. Dessa delar är:

1. **Variabler och operationer:** beskriver olika typer av variabler och de olika aritmetiska och logiska operationer som kan göras på dessa.
2. **Kommando:** beskriver de kommandon som används för att arbeta med programmen, editera, spara och radera dem.
3. **Programinstruktioner:** beskriver de kommandon som används i programmen.
4. **Funktioner:** beskriver sträng och numeriska utskriftsfunktioner.

I varje sektion finner du kommandona listade i bokstavsordning. En komplett beskrivning av alla BASIC-kommando finner du i VIC's programmeringshandbok. Den finns där du köpte din VIC.

1. VARIABLER OCH OPERATIONER

a. Variabler

VIC använder tre typer av variabler i BASIC. Dessa är: flyttal, heltal och strängvariabler.

FLYTTAL är numeriska variabler inom området 10E-38 till 10E+38 med en noggrannhet av nio siffror. När talet blir längre än 9 siffror skrivs det u i exponentform. Ett tal med en siffra och 8 decimaler följt av ett E och ett tal som är 10-exponenten av talet. Till exempel, kommer talet 123456789001 att skrivas som 1.2345689+11.

HELTAL är variabler för tal inom intervallet —32768 till 32767, utan decimaler. En heltalsvariabel kräver mindre plats i minnet. Skillnaden är inte värd att lägga märke till om du inte har stora mängder med variabler i så kallade matriser. Exempel på heltal är 5, 10 och —103.

STRÄNGAR används för att lagra tecken av olika slag, bokstäver, siffror eller andra tecken som VIC kan skriva ut. Ett exempel på en strängvariabel är "VIC 20".

Ett variabelnamn kan bestå av en bokstav, två bokstäver eller en bokstav följt av en siffra.

En heltalsvariabel specificeras genom att variabelnamnet följs av ett %-tecken. Strängvariabler har ett $-tecken som sista tecken i namnet.

EXEMPEL:
Namn på flyttal: A, A5, BZ
Namn på heltal: A%, A5%, BZ%
Namn på strängar: A$, A5$, BZ$

MATRISER används för att samla ihop en större mängd med variabler under samma namn. Detta gör man genom att använda ett antal extra siffror i variabelnamnet. Vilken typ av variabel och hur många de får finnas anges i DIM-kommandot. Alla typer av variabler kan användas. Efter DIM-kommandot anges variabelnamnen och inom parentes hur många variabler de får innehålla.

EXEMPEL: DIM A(7), BZ%(11), A$(87)

Matriserna kan innehålla mer än en dimension. En tvådimensionell matris kan ses som en tabell med rader och kolumner där första siffran innanför parentesen betyder raden och den andra kolumnen.

EXEMPEL: DIM A(7,2), BZ%(2,3,4), Z$(3,2)

Det finns tre stycken variabelnamn som är reserverade för VIC's interna bruk och kan normalt inte användas. Dessa variabler är ST, TI och TIS. ST är en status(läges)variabel för kommandon som berör in/ut-operationer. Värdet på ST kommer till exempel att ändra sig när det är problem med en programladdning från kassettband eller flexskivminne. En mera detaljerad beskrivning av ST finns i programmeringshandboken för VIC.

TI OCH TIS är variabler som berör den inbyggda klocka som finns i VIC. Variabeln TI ökas med 1 varje 1/60 sekund. Den börjar på noll när din VIC slås på. Värdet kan endast ändras genom att förändra värdet i TIS.

TIS är en sträng som kontinuerligt uppdateras av systemet. De första två tecknen innehåller antalet timmar, det tredje och fjärde antalet minuter och det femte och sjätte antalet sekunder. Denna variabel kan ges ett annat värde (så länge tecknen är siffror) och kommer att uppdateras från detta tal automatiskt.

EXEMPEL: TIS = "101530" ställer klockan till 10:15:30

Klockan nollställs när VIC slås på.

b. OPERATIONER

De aritmetiska operationerna innehåller följande tecken:

+ addition
— subtraktion
* multiplikation
/ division
↑ exponering

På en rad som innehåller flera operationer, utförs operationerna i en bestämd ordning. Om flera operatorer används tillsammans ger datorn följande prioritetsordning på dem: Först exponering, sedan multiplikation och division, och sist addition och subtraktion. Om du vill att operationerna skall ske i en viss ordning, tillåter VIC BASIC att du isolerar en beräkning genom att omge den med parenteser. Operationer inom parentes görs först av alla. Var noga med att formeln innehåller lika många vänsterparenteser som högerparenteser, annars får du ett "SYNTAX ERROR" när du kör programmet.

Det finns också operatorer för att jämföra två värden:

=	lika med
<	mindre än
>	större än
< = eller = <	mindre eller lika med
> = eller = >	större eller lika med
<> eller <>	inte lika med

Till slut så finns det tre logiska operatorer:

AND

OR

NOT

Dessa används oftast för att lägga ihop flera villkor i ett IF...THEN-kommando.

EXEMPEL:
IF A=B AND C=D THEN 100 både A=B och C=D skall vara sanna.

IF A=B OR C=D THEN 100 någon av A=B eller C=D skall vara sann.

2. KOMMANDON

CONT (continue = fortsätt)

Kommandot används för att starta upp ett program där programmet blev avbrutet t ex genom att STOP-tangenten trycktes, ett STOP eller END-kommando i programmet. Programmet körs igång på samma ställe där det blev avbrutet.

CONT fungerar inte om du ändrat något i programmet inte ens om du bara flyttat markören upp till en rad och tryckt RETURN utan att ändra någonting i raden. Inte ens om programmet stannades genom ett fel, eller om du gör ett fel innan du kör igång programmet med CONT. I dessa fall svarar VIC med "CAN'T CONTINUE ERROR".

LIST

LIST-kommandot tillåter att du tittar på de programrader som du skrivit eller laddat in i VIC's minne. Skriver du LIST-kommandot utan några siffror kommer hela programmet att listas ut på din skärm (Listningshastigheten kan sänkas genom att trycka på CTRL eller stoppas genom att trycka på STOP-tangenten.) Om LIST följs av ett radnummer listas endast raden med det numret ut. Anges två nummer med ett streck emellan listas alla rader mellan dessa två nummer ut. Skriver du LIST med ett nummer följt av ett streck listas raderna från och med denna rad till programmets slut. Och skriver du LIST med ett streck och ett nummer listas alla rader till och med det numret. Med dessa variationer på kommandot kan du lista vilken del av programmet du vill och få fram raderna på skärmen för att göra ändringar.

EXEMPEL:

LIST	Listar hela programmet.
LIST 10—	Listar rad 10 till programmets slut.
LIST 10	Listar rad 10.
LIST —10	Listar programmet från början till rad 10.
LIST 10—20	Listar rad 10 till 20.

LOAD

Detta kommando används när du har ett program på kassettband eller flexskiva och vill hämta programmet för att köra det. Om du bara skriver LOAD och trycker RETURN, kommer VIC att ladda in det första program den hittar på kassettbandet. Programmet kan du sedan köra, lista eller göra vad du vill med. Du kan också skriva LOAD följt av ett programnamn mellan citationstecken. Efter citationstecknet kan du lägga till ett kommatecken och ett tal eller en numerisk variabel som visar varifrån programmet skall ladda. Om inget nummer anges antar VIC att den skall ladda från enhet#1, kassettenheten.

En annan vanlig enhet som program laddas ifrån är flexskivminnet, enhet #8.

EXEMPEL:

LOAD	Läser in första programmet från kassettbandet.
LOAD "HEJSAN"	Söker efter programmet med namnet "HEJSAN" på kassettbandet och läser in det om den hittar det.
LOAD A$	Söker efter programmet med namnet som finns i AS och läser in det.

LOAD "HEJSAN",8 Söker efter programmet med namnet "HEJSAN" på flexskiveminnet.

LOAD "*",8 Laddar in det första programmet på flexskivan.

LOAD-kommandot kan användas i ett program för att ladda in och köra nästa program på bandet.

NEW

Detta kommando raderar hela programmet som ligger i VIC's minne och nollställer alla variabler som har använts. Om du inte tidigare sparat undan programmet på kassettband eller flexskiva så är ditt program borta för evigt om du inte skriver in det igen. VAR FÖRSIKTIG när du använder detta kommando!

NEW-kommandot kan också användas i programmet. När VIC kommer fram till detta kommando raderar den allt som finns i minnet och stannar. Detta är bra om man vill lämna allt städat och klart när ett program har körts.

RUN

När du har ett program i minnet och vill köra det, skriver du RUN och VIC börjar utföra alla kommandon i programmet. Om inget radnummer anges börjar VIC på raden med det lägsta numret. Om numret finns börjar körningen på det radnumret.

EXEMPEL:

RUN Programmet kör igång från första raden.

RUN 100 Programmet kör igång i rad 100.

RUN X SYNTAX ERROR (Du måste alltid skriva ut numret och inte använda bokstäver).

SAVE

Detta kommando sparar undan det program som för tillfället finns i VIC's minne på kassettband eller flexskiva. Skriver du bara SAVE försöker VIC att spara programmet på kassettbandet. Det finns ingen kontroll att bandet redan innehåller ett program, så var försiktig så inget program går förlorat när du sparar ditt program. Om du anger ett namn efter SAVE-kommandot så skrivs det namnet in på bandet eller flexskivan. Det blir då enklare att hitta programmet när du skall hämta det igen. Efter namnet, mellan citationstecknen, kan du lägga till ett kommatecken och ett nummer eller en variabel. Numret talar om på vilken enhet som programmet skall sparas. Enhet nummer 1 är kassettenheten och nummer 8 är flexskivminnet. Efter det första numret kan det läggas ett kommatecken och ett nummer till, som kan vara 1 eller 2. Om numret är en 2:a skrivs en slut på bandet markering (end of tape) in på bandet. Om VIC söker efter ett program och finner denna markering så skrivs "FILE NOT FOUND ERROR" ut.

EXEMPEL:

SAVE	Sparar programmet på kassettband utan namn.
SAVE "HEJSAN"	Sparar programmet med namnet HEJSAN på bandet.
SAVE A$	Sparar programmet med namnet som finns i A$.
SAVE "HEJSAN",8	Sparar programmet med namnet HEJSAN på flexskivan.
SAVE "HEJSAN",1,2	Sparar programmet med namnet HEJSAN och lägger in en "end of tape" markering på bandet.

VERIFY

Detta kommando gör att VIC går tillbaka och jämför programmet på bandet eller flexskivan med det i minnet. Har programmet utförts utan klagomål så är det ett bevis på att ditt program blev sparat korrekt. Det kunde ha varit ett dåligt band du sparade programmet på. Detta kommando är bra om du vill lägga in ett program på första tomma position på bandet. Vad du gör är att säga åt VIC att jämföra det sista programmet med det i minnet. Detta kommer den att göra och upptäcka att programmen inte stämde överens, vilket du redan visste. Nu är bandet där du vill ha det. Nu kan du spara nästa program utan någon risk att förstöra något annat program.

Enbart VERIFY gör att VIC jämför det första program den hittar på bandet, oavsett namn, med det som finns i minnet. VERIFY med ett namn eller citationstecken söker efter programmet med det namnet för att jämföra det med minnet. Dessutom kan du lägga till ett kommatecken och ett nummer eller en variabel som talar om för VIC vilken enhet den skall söka på (1 för kassettband, 8 för flexskiva).

EXEMPEL:

VERIFY	Jämför med nästa program på band.
VERIFY "HEJSAN"	Söker efter "HEJSAN" på band, och jämför det med minnet.
VERIFY "HEJSAN",8	Söker efter "HEJSAN" på flexskiva, och jämför det med minnet.

3. PROGRAMINSTRUKTIONER

CLOSE

Detta kommando stänger en fil som du tidigare öppnat med OPEN-kommandot. Det nummer som följer efter kommandot är det filnummer som skall stängas.

EXEMPEL:

CLOSE 2 Endast fil #2 stängs.

CLR

Kommandot raderar alla variabler som finns i minnet, men lämnar programmet intakt. Detta kommando utförs automatiskt när du kör i gång programmet (RUN).

CMD

Ändrar den normala utenheten (skärmen) till en annan. Alla PRINT- och LIST-utskrifter kommer istället att göras på den angivna enheten. Denna enhet kan vara en skrivare, kassettenhet, flexskivminne eller något annat. Innan CMD-kommandot utföres måste en fil öppnas till enhet (OPEN). I CMD-kommandot skall anges det filnummer där utskriften skall gå.

EXEMPEL:

OPEN 1,4 Öppnar en fil till enhet #4 (skrivare).

CMD 1 All normal utskrift går till skrivaren.

LIST Listningen av programmet går ut till skrivaren, inte skärmen — till och med ordet LIST.

För att få tillbaka utskriften till skärmen, stäng bara filen.

DATA

Kommandot följs av en lista med data som används av kommandot READ. Datat kan vara numeriska variabler eller text och separeras med kommatecken. Text behöver inte omges av citationstecken, om inte texten innehåller något av följande tecken: Blanktecken, kolon, kommatecken. Om det finns två kommatecken utan någonting mellan dem, tas det som värdet 0 eller en tom textsträng.

EXEMPEL:

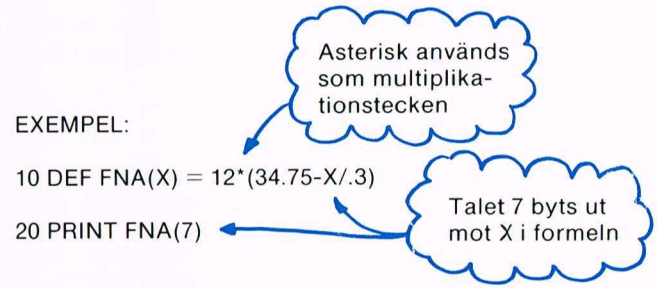
DATA 100,200,FRED,"HEJ MAMMA",,3.14,ABC123

Eftersom DATA-kommandot inte behöver utföras för att datat skall bli läsbart, så är det en god idé att lägga alla DATA-kommandon sist i programmet. Detta gör att ditt program går fortare att köra.

DEF FN

Med detta kommando kan du definiera en komplex funktion som du sedan kan anropa i programmet på ett enkelt sätt. Så här kan mycket plats sparas i programmet om det innehåller samma beräkning flera gånger.

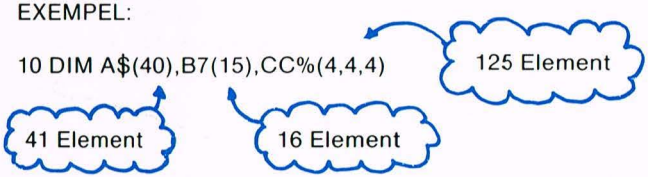
Funktionens namn blir FN plus ett variabel namn (1 eller 2 tecken). Först måste funktionen definieras genom att du skriver DEF följt av funktionsnamnet. Efter namnet följer ett par med parenteser med en variabel innanför (i exemplet ett X). Därefter kommer likhetstecknet och formeln för funktionen. När du vill använda funktionen byter du ut X mot det aktuella värdet, som i rad 20 i exemplet nedan:



DIM

Innan du kan använda matrisvariabler som har fler än 11 variabler, måste ett DIM-kommando utföras för den matrisvariabeln. DIM-kommandot följs av variabelnamnet på matrisen (ett normalt variabelnamn, 1 eller 2 tecken). Efter namnet kommer två parenteser med värden (eller numeriska variabler) innanför, som betecknar antalet variabler i varje dimension. Antalet dimensoner är inte begränsat, men kom ihåg att minnesutrymmet inte räcker till för hur stora matriser som helst. För att beräkna antalet variabler som ett DIM-kommando skapar så multiplicerar du antalet element i varje dimension i matrisen.

EXEMPEL:



Du kan dimensionera fler matriser i samma DIM-kommando genom att separera dem med kommatecken. Om programmet försöker att utföra ett DIM-kommando på samma variabel mer än en gång, får du ett felmeddelande. Det är bäst att placera DIM-kommandot i början på programmet.

END

När VIC kommer till detta kommando på en rad slutar den att köra programmet precis som om det vore sista raden i programmet. Du kan använda CONT-kommandot för att starta upp programmet igen.

FOR...TO...STEP

Detta kommando används tillsammans med NEXT för att lägga upp en sektion i programmet som skall repeteras ett antal gånger (en loop). Du vill kanske bara att VIC räknar upp till ett högt tal, för att få den att vänta ett tag (tidsfördröjning), eller du kanske vill räkna någonting. Detta är det mest förekommande BASIC-kommandot i programmen.

Uppställningen för kommandot är följande:

FOR (loopvariabel) = (startvärde) TO (slutvärde). Loopvariabel är en helt vanlig variabel som kan ändras under pågående loop. Start och slutvärdet är gränserna för loopvariabeln.

Följande händer vid FOR-kommandot. Loopvariabel tilldelas värdet "startvärde". Slutvärdet lagras undan i VIC för senare referens. När VIC kommer fram till ett NEXT-kommando, ökar den värdet av loopvariabeln med 1. Variabeln jämförs sedan med slutvärdet. Om loopvariabeln inte är större än slutvärdet hoppar VIC tillbaka till raden efter FOR-kommandot. Är loopvariabeln större fortsätter VIC med nästa rad efter NEXT-kommandot.

EXEMPEL:

```
10 FOR L = 1 TO 10

20 PRINT L

30 NEXT L

40 PRINT "FÄÄÄÄRDIG! L="L
```

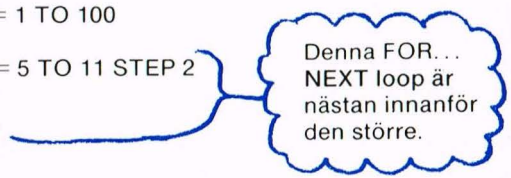
Detta program skriver ut talen från 1 till 10 på skärmen följt av meddelandet "FÄÄÄÄRDIG! L=11". Förstår du hur det fungerar? Om inte, läs då igenom texten och följ med i programmet steg för steg.

Slutvärdet kan följas av ordet STEP och ett värde eller en variabel. I detta fall kommer VIC att använda värdet och addera till loopvariabeln vid NEXT-kommandot (istället för 1). Detta gör det möjligt att räkna baklänges eller med decimaler.

Du kan lägga upp loopar i en annan loop. Detta kallas för nästade loopar. Du skall vara försiktig vid nästade loopar och komma ihåg att den första loopen slutar sist.

EXEMPEL:

```
10 FOR L = 1 TO 100
20 FOR A = 5 TO 11 STEP 2
30 NEXT A
40 NEXT L
```



Felaktigt:

```
10 FOR L = 1 TO 100
20 FOR A = 5 TO 11 STEP 2
40 NEXT L
30 NEXT A
```

GET

GET-kommandot används för att läsa data från tangentbordet, ett tecken i taget. När GET-kommandot utföres läses det tecken som trycks på tangentbordet in. Har ingen tangent tryckts ned, fås ett "tomtecken" tillbaks. RETURN-tangenten behöver inte tryckas ned för att inläsning skall ske (jämför INPUT). RETURN kan också läsas av.

Efter ordet GET så följer ett variabelnamn, vanligen en strängvariabel. Används en numerisk variabel kommer VIC att svara med ett felmeddelande (om någon annan än en siffertangent trycks ned). GET används mest för att konstruera en loop som väntar på att en tangent skall tryckas ned.

EXEMPEL:

```
10 GET AS: IF AS = "" THEN 10
```

GET#

Används tillsammans med en tidigare öppnad fil (OPEN) för att läsa in ett tecken i taget.

EXEMPEL:

```
GET#1,A$
```

GOSUB

Detta kommando gör likadant som GOTO men den kommer dessutom ihåg var den kom ifrån. När VIC kommer till en rad med kommandot RETURN, så hoppar den tillbaka till raden efter det senaste GOSUB-kommandot. Detta är användbart om du vill ha samma programsektion utförd på flera ställen i programmet. Istället för att upprepa programsektionen på varje ställe så läggs GOSUB-kommandot in med ett efterföljande radnummer, som visar var programsektionen ligger. GOSUB 800 betyder att VIC hoppar till rad 800 och fortsätter programkörningen där.

GOTO eller GO TO

När VIC kommer till ett GOTO-kommando, hoppar den till raden som anges efter kommandot.

IF...THEN

IF...THEN-kommandot gör det möjligt för VIC att analysera en situation och välja en av två möjliga vägar. Om uttrycket som utförs blir sant, utförs ordet som står efter THEN. Ordet kan vara ett tal som då VIC hoppar till. Det kan också vara ett kommando av något slag. Om uttrycket är falskt fortsätter VIC på nästa programrad (inte nästa kommando på raden).

Uttrycket som undersöks kan vara en variabel eller en formel, där sant är skilt från noll (<>0) och falskt lika med noll (=0). Ofta används relationsoperanderna i uttrycken (= , < , > , AND , OR , NOT). Om uttrycket blir sant blir värdet —1 och falskt 0. Se kapitlet om relationsoperatorer och hur de fungerar.

INPUT

INPUT-kommandot gör det möjligt för VIC att låta personen som sitter vid tangentbordet skriva in värdet på en variabel. Programmet stannar, skriver ut ett frågetecken och väntar på att personen i fråga skall skriva in svaret och trycka på RETURN-tangenten.

Kommandot följs av ett eller flera variabler separerade med kommatecken. Det kan också förekomma ett meddelande mellan citationstecknen innan variablerna. Detta meddelande (eng. promt) måste följas av ett semikolon efter sista citationstecknet. Om fler variabler skall skrivas in måste de separeras med komman vid inskriften.

EXEMPEL:

```
10 INPUT "SKRIV ETT VÄRDE PÅ A ";A

20 INPUT "OCH DITT NAMN TACK ";A$

30 INPUT B$

40 PRINT "NU HADE DU INGEN AAANING OM VAD JAG VILLE VETA"
```

INPUT#

Fungerar som INPUT men tar datat från en tidigare öppnad (OPEN) fil.

LET

Ordet LET används mycket sällan i program, eftersom det inte har någon praktisk btydelse, men det är faktiskt grunden till alla BASIC-program. Variabeln som skall få resultatet av beräkningen finns på vänstra sidan av likhetstecknet medan formeln och värdena finns på den högra.

EXEMPEL:

```
10 LET A = 5

20 B = 6

30 C = A*B+3

40 DS = "HEJSAN"
```

NEXT

NEXT-kommandot används alltid tillsammans med FOR-kommandot. När programmet kommer till ett NEXT-kommando går den tillbaka till FOR-kommandot för att kontrollera loopen. (Se FOR-kommandot för mera information.) Om loopen är slut fortsätter programmet till nästa kommando efter NEXT. NEXT kan följas av ett variabelnamn, eller flera variabelnamn separerade med kommatecken. Om inget variabelnamn anges utförs den senast påbörjade loopen. Om flera variabler anges utförs de från vänster till höger.

EXEMPEL:

```
10 FOR L = 1 TO 10:NEXT

20 FOR L = 1 TO 10:NEXT L

30 FOR L = 1 TO 10:FOR M = 1 TO 10:NEXT M,L
```

ON

Detta kommando gör GOTO och GOSUB till specialvarianter av IF-kommandot. Ordet ON följs av en formel som ger ett värde. Efter ordet GOTO eller GOSUB följer en lista med radnummer skilda åt med kommatecken. Om resultatet av beräkningen blir 1, hoppar programmet till det första radnumret i listan. Är resultatet 2 hoppar den till det 2:a radnumret. Om resultatet är negativt, 0 eller större än antalet radnummer i listan, hoppar programmet till raden efter ON-kommandot.

EXEMPEL:

```
10 INPUT X

20 ON X GOTO 10,50,50,50

30 PRINT "NäHä"

40 GOTO 10

50 PRINT "HÄPP!"

60 ON X GOTO 10,30,30
```

OPEN

OPEN-kommandot gör att VIC kan kommunicera med andra enheter såsom kassettenhet, flexskivminne, skrivare eller skärmen. Ordet OPEN följs av ett nummer som övriga BASIC-kommando refererar till. Detta nummer är mellan 1 och 255. Efter det första numret kommer ett andra separat med kommatecken. Detta är enhetsnumret, 0 för tangentbordet 1 för kassettenheten, 2 för RS232, 3 för bildskärm, 4 för skrivare och 8 för flexskivminne. Det är bäst att använda samma filnummer som enhetsnummer, det blir lättare att hålla reda på filerna då. Efter det andra numret kan ett kommatecken och ett tredje nummer förekomma, vilket är sekundäradressen. Detta betyder för kassettenheten om den skall skriva eller läsa. 0 för läsning, 1 för skrivning och 2 för skrivning med "end of tape" markering på slutet. Är enheten ett flexskivminne så betyder numret ett visst buffer eller kanalnummer. I skrivaren betyder numret olika kommandon. (Se VIC programmeringshandbok för mera detaljer om detta.) Efter det tredje numret kan det dessutom finnas en sträng med text som ger ett kommando till flexskivminnet eller ger ett programnamn på kassetten.

EXEMPEL:

10 OPEN 1,0	Öppnar en fil till skärmen.
20 OPEN 2,1,0,"D"	Öppnar en fil för läsning på kassettenheten. Filen som den skall söka efter heter "D".
30 OPEN 3,4	Öppnar en fil till skrivaren.
40 OPEN 4,8,15	Öppnar kommandokanalen till skivminnet.

Se också: CLOSE, CMD, GET#, INPUT#, PRINT#, systemvariabeln ST och appendix B.

POKE

POKE-kommandot följs alltid av två stycken nummer eller formler. Det första numret är positionen i VIC's minne. Positionsnumret kan vara

mellan 0 och 65000. En del av positionerna kan du använda till att ändra på färg eller ljud som du sett i tidigare kapitel. En del positioner använder VIC för att hålla reda på programmet och så vidare. Om du experimenterar med POKE-kommandot kan du säkert få en del intressanta effekter. Om någonting händer och du inte kan stoppa det, är det bara att stänga av VIC eller trycka på STOP och RESTORE.

Det andra numret har ett värde mellan 0 och 255, och placeras i den position som angavs i det första numret.

EXEMPEL:

```
10 POKE 36879,8

20 POKE 9*16  3 + 15.27
```

PRINT

PRINT-kommandot är det första kommando som de flesta lär sig att använda, men det finns många finesser med kommandot som få lär sig använda. I PRINT-kommandot finns följande saker:

Text mellan citationstecken
Variabelnamn
Funktioner
Formatteringstecken

Tecken innanför citationstecken skrivs ut som de är. Namn utanför citationstecken tas för ett variabelnamn och värdet på den variabeln skrivs ut. Funktionernas värden kommer också att skrivas ut. Formatteringstecken används för att få en snygg utskrift. Kommat används för att dela in skärmen i två delar, medan semikolon inte ger några blanktecken alls mellan utskrifterna. Båda tecknen kan användas sist i kommandot. Detta får då effekten att nästa utskrift skrivs ut som om den fanns på samma rad.

EXEMPEL:

```
10 PRINT "HEJSAN"

20 PRINT "HEJSAN, "A$

30 PRINT A+B;

50 PRINT J;

60 A,B,C,D
```

Se också: POS(), SPC(), TAB() funktionerna.

PRINT#

Det finns några skillnader mellan det vanliga PRINT-kommandot och PRINT#-kommandot. För det första följs kommandot av ett nummer som refererar till en fil som du nyss öppnade (OPEN) till någon enhet. Efter numret kommer ett kommatecken och en lista med de saker som skall skrivas ut. Kommatecknet och semikolonet har samma effekt, de adderar blanktecken. TAB och SPC kan på vissa enheter fungera dåligt.

EXEMPEL:

```
100 PRINT#1,"HEJSAN DU!";A$,B$
```

READ

Detta kommando används för att läsa information från rader med DATA-kommandon och lägga in information i variabler. Du bör se upp med att försöka läsa in text när READ-kommandot vill ha ett numeriskt värde. Om så sker svarar VIC med "TYPE MISMATCH ERROR".

REM

REMark (anmärkning) används för att lägga in kommentarer i programmet. Detta gör att den som läser programmet har lättare att förstå vad programmeraren menar med sina kommandon. REM-kommandot påverkar inte på något sätt programmet utom att det blir större. Texten efter kommandot kan vara vilken text som helst, men grafiska tecken ger konstiga resultat. (Se Programmerarens handbok för mera information).

RESTORE

Den pekare som anger vilket objekt i DATA-kommandot som ligger på tur för att läsas av READ-kommandot, nollställs till det första objektet när detta kommando utförs. Detta gör det möjligt att läsa om informationen från början. Kommandot RESTORE står ensamt på raden utan extra text.

RETURN

Detta kommando används alltid tillsammans med GOSUB-kommandot. När VIC kommer till ett RETURN-kommando hoppar den tillbaka till första kommandot efter GOSUB. Om inget GOSUB-kommando fanns före, svarar VIC med "RETURN WITHOUT GOSUB ERROR". Ingen extra text på RETURN.

STOP

Detta kommando stannar programmet. VIC skriver ut, "BREAK IN LINE xxxx", där xxxx är raden som innehåller STOP-kommandot. Med CONT-kommandot kan programmet startas om igen där det stannade. STOP-kommandot används för att leta fel i program.

SYS

Kommandot följs av ett nummer eller en numerisk variabel i området 0-65535. När VIC får detta kommando hoppar den ner i minnet och utför det maskinspråksprogram som finns på den angivna adressen. Det påminner om USR-funktionen, men den kan inte överföra parametrar.

WAIT

Kommandot används för att stanna programmet till dess en viss minnesposition ändrar sitt värde på ett specificerat sätt. Ordet WAIT följs av ett nummer som är den position som skall kontrolleras. Därefter kommer ett kommatecken och ett nummer. Det kan även finnas ännu ett kommatecken och ett tredje nummer. Dessa två sista nummer måste vara inom området 0-255.

På innehållet i minnespositionen görs först en exklusive-OR med det tredje numret om det finns, sedan görs AND med det andra numret. Om resultatet är 0 går VIC tillbaka och kontrollerar innehållet igen. Är resultatet inte 0 fortsätter programmet till nästa kommando.

4. FUNKTIONER

a. NUMERISK

ABS(X)

Du får tillbaka absolutvärdet av talet, utan tecken. Resultatet är alltid positivt.

ATN(X)

Ger vinkeln i radianer, vars tangent är X.

COS(X)

Ger cosinus för vinkel X i radianer.

EXP(X)

Ger värdet av den matematiska konstanten e (2.71827183) upphöjt till X.

FNXX(X)

Ger värdet av den användardefinierade funktionen XX skapad genom FNXX-kommndot.

INT(X)

Ger ett trunkerat värde av X, det vill säga utan decimaler. Resultatet blir alltid mindre eller lika med värdet av X. Detta gäller även negativa tal.

Om INT-funktionen används för att avrunda uppåt eller nedåt ser formeln ut så här:

EXEMPEL:

X = INT(X*100+.5)/100 Avrundar till närmaste öre.

LOG(X)

Ger den naturliga logaritmen av X. Naturliga logaritmen har basen e (Se EXP(X)). För att konvertera till bas 10 så dividerar du med LOG(10).

PEEK(X)

Detta används för att ta reda på innehållet i minnespositionen X (X=0-65535). Resultatet blir -255. Denna funktion används ofta tillsammans med POKE-kommandot.

RND(X)

Denna funktion ger som resultat ett slumptal mellan 0 och 1. Detta är användbart i spel, där du vill simulera tärningskast eller olika utfall. Det kan också användas i vissa statiska sammanhang. Det första slumptalet bör genereras genom formeln RND(-TI), så att det första värdet blir olika varje gång. Efter detta bör värdet X vara 1 eller något positivt värde. Om X är 0 blir resultatet samma som föregående slumptal. Ett negativt värde sätter ett nytt "frö" i slumpgeneratorn. Användningen av samma negativa tal ger samma sekvens av slumptal.

För att simulera ett tärningskast använd formeln INT(RND(1)*6+1). Först multipliceras slumptalet från 0-1 med 6 och intervallet utökas till 0-6 (större än 0 och mindre än 6). Addering med 1 ger 1-7. INT-funktionen tar bort alla decimaler och kvar finns ett heltal mellan 1 och 6.

För att simulera 2 tärningar adderar du värdet från två stycken slumptal enligt ovanstående formel.

EXEMPEL:

100 X = INT(RND(1)*6)+INT(RND(1)*6)+2	Simulerar två tärningar.
100 X = INT(RND(1)*1000)+1	Ett heltal från 1-1000.
100 X = INT(RND(1)*150)+100	Ett heltal från 100-249.

SGN(X)

Ger tecknet på värdet X, om det är positivt, negativt eller lika med 0. Resultatet blir +1 om positivt, 0 om noll och —1 om negativt.

SIN(X)

Ger sinus av vinkeln X. Vinkeln X anges i radianer.

SQR(X)

Ger kvadratroten ur värdet X. Där X är positivt. Om X är negativt svarar VIC med "ILLEGAL QANTITY ERROR".

TAN(X)

Ger tangenten av X där X är vinkeln i radianer.

USR(X)

När denna funktion används hoppar VIC till ett maskinspråksprogram vars startadress finns i minnespositionerna 1 och 2. Parametern X överförs till maskinspråksprogrammet, vilken kommer att ge tillbaka ett annat värde. Se VIC PROGRAMMERINGSHANDBOK för mera detaljer om detta och maskinspråksprogram.

b. STRÄNGFUNKTIONER

ASC(X$)

Funktionen ger som resultat ASCII-koden för första tecknet i X$.

CHR$(X)

Detta är motsatsen till ASC, den ger tecknet vars ASCII-kod är X.

LEFT$(X$,X)

Ger X stycken tecken av den vänstra delen av X$.

LEN(X$)

Ger antal tecken i X$ (inkluderar blanktecken och symboler).

MID$(X$,S,X)

Ger som resultat X stycken tecken ur strängen X$ med början i det S:te tecknet.

RIGHT$(X$,X)

Ger X stycken tecken av den högra delen av X$.

STR$(X)

Ger en sträng som motsvarar den utskrift du får av variabeln X.

VAL(X$)

Denna funktion är motsatsen till STR$. Funktionen omvandlar en sträng till ett numeriskt värde. Strängen söks av från vänster till höger så länge den innehåller numeriskt format. Upptäcker VIC ett tecken som inte hör dit slutar den att konvertera.

EXEMPEL:

10 X = VAL("123.456")	X = 123.456
10 X = VAL("12A13B")	X = 12
10 X = VAL("RIUO17*")	X = 0
10 X = VAL("—1.23.23.23")	X = 1.23

c. ÖVRIGA FUNKTIONER

FRE(X)

Denna funktion ger antalet lediga minnespositioner, oavsett värdet på X.

POS(X)

Funktionen visar i vilken kolumn (0-21) nästa utskrift kommer att börja. X kan vara vilket värde som helst och används inte.

SPC(X)

Används i PRINT-kommandot för att skriva ut X stycken blanktecken.

TAB(X)

Används i PRINT-kommandot för att påbörja nästa utskrift i kolumnen X.

BILAGA D:
FÖRKORTNINGAR FÖR BASIC-ORD

För att spara tid när du skriver in programkommandon tillåter VIC att du förkortar de flesta orden. Förkortningen för PRINT är ett frågetecken. Förkortningen för de övriga orden görs genom att skriva första eller de två första tecknen i ordet och sedan hålla SHIFT nedtryckt medan du trycker det efterföljande tecknet. När förkortningen används i programrader kommer den att skrivas ut i sin längre form när programmet listas. Notera att en del ord innehåller den första parentesen, andra gör det inte.

Kommando	Förkortning	Ser ut så här på skärmen
AND	A SHIFT N	A
NOT	N SHIFT O	N
CLOSE	CL SHIFT O	CL
CLR	C SHIFT L	C
CMD	C SHIFT M	C
CONT	C SHIFT O	C
DATA	D SHIFT A	D
DEF	D SHIFT E	D
DIM	D SHIFT I	D
END	E SHIFT N	E
FOR	F SHIFT O	F
GET	G SHIFT E	G
GOSUB	GO SHIFT S	GO
GOTO	G SHIFT O	G
INPUT#	I SHIFT N	I
LET	L SHIFT E	L
LIST	L SHIFT I	L
LOAD	L SHIFT O	L
NEXT	N SHIFT E	N
OPEN	O SHIFT P	O
POKE	P SHIFT O	P
PRINT	?	?

Kommando	Förkortning	Ser ut så här på skärmen
PRINT#	P SHIFT R	P
READ	R SHIFT E	R
RESTORE	RE SHIFT S	RE
RETURN	RE SHIFT T	RE
RUN	R SHIFT U	R
SAVE	S SHIFT A	S
STEP	ST SHIFT E	ST
STOP	S SHIFT T	S
SYS	S SHIFT Y	S
THEN	T SHIFT H	T
VERIFY	V SHIFT E	V
WAIT	W SHIFT A	W
ABS	A SHIFT B	A
ASC	A SHIFT S	A
ATN	A SHIFT T	A
CHR$	C SHIFT H	C
EXP	E SHIFT X	E
FRE	F SHIFT R	F
LEFT$	LE SHIFT F	LE
MID$	M SHIFT I	M
PEEK	P SHIFT E	P
RIGHT$	R SHIFT I	R

Kommando	Förkortning	Ser ut så här på skärmen
RND	R SHIFT N	R
SGN	S SHIFT G	S
SIN	S SHIFT I	S
SPC (	S SHIFT P	S
SQR	S SHIFT Q	S

Kommando	Förkortning	Ser ut så här på skärmen
STR$	ST SHIFT R	ST
TAB (	T SHIFT A	T
USR	U SHIFT S	U
VAL	V SHIFT A	V

BILAGA E: BAKGRUNDS- OCH RAMKOMBINATIONER I FÄRG

Du kan ändra färgen på bakgrunden eller ramen när du vill, genom att skriva:

POKE 36879,X

Där X är ett nummer ur tabellen nedan. POKE 36879,27 ställer om färgrna till de normala, CYAN som ramfärg och VIT som bakgrund.
Försök med att skriva POKE 36879,8.

Tryck sedan **CTRL** 2 WHT och du får vita

bokstäver på svart bakgrund. POKE-kommandot är ett snabbt och enkelt sätt att ändra färgen på skärmen.

	RAM							
SKÄRM	**SVART**	**VIT**	**RÖD**	**CYAN**	**VIOLETT**	**GRÖN**	**BLÅ**	**GUL**
SVART	8	9	10	11	12	13	14	15
VIT	24	25	26	27	28	29	30	31
RÖD	40	41	42	43	44	45	46	47
CYAN	56	57	58	59	60	61	62	63
VIOLETT	72	73	74	75	76	77	78	79
GRÖN	88	89	90	91	92	93	94	95
BLÅ	104	105	106	107	108	109	110	111
GUL	120	121	122	123	124	125	126	127
ORANGE	136	137	138	139	140	141	142	143
LJ. ORANGE	152	153	154	155	156	157	158	159
SKÄR	168	169	170	171	172	173	174	175
LJ. CYAN	184	185	186	187	188	189	190	191
LJ. VIOLETT	200	201	202	203	204	205	206	207
LJ. GRÖN	216	217	218	219	220	221	222	223
LJ. BLÅ	232	233	234	235	236	237	238	239
LJ. GUL	248	249	250	251	252	253	254	255

BILAGA F
TABELL ÖVER MUSIKTONER

TON	VÄRDE	TON	VÄRDE
C	135	G	215
C#	143	G#	217
D	147	A	219
D#	151	A#	221
E	159	B	223
F	163	C	225
F#	167	C#	227
G	175	D	228
G#	179	D#	229
A	183	E	231
A#	187	F	232
B	191	F#	233
C	195	G	235
C#	199	G#	236
D	201	A	237
D#	203	A#	238
E	207	B	239
F	209	C	240
F#	212	C#	241

LJUDKOMMANDO:	DÄR X ÄR MELLAN:	FUNKTION:
POKE 36878, X	0 och 15	volym
POKE 36874, X	128 och 255	ton
POKE 36875, X	128 och 255	ton
POKE 36876, X	128 och 255	ton
POKE 36877, X	128 och 255	brus

BILAGA G:
20 LJUDEFFEKTER TILL VIC 20

Här följer några exempel på programrutiner som kan hjälpa dig att få olika ljudeffekter i dina program. Du kan skriva in dem i din VIC 20 precis som de är eller lägga in dem i större program. Detta är givetvis inte alla ljudeffekter som VIC kan åstadkomma, så känn dig fri och kreativ att skapa nya effekter.

De program som listats här får effekten att ingenting annat kan göras under det att ljudet låter. Det finns ett sätt som gör att programmets övriga arbete inte avstannar. Detta beskrivs mera detaljerat i PROGRAMMERINGSHANDBOKEN.

Glöm inte att lägga till radnummer när du skriver in programmen in din VIC 20. Radnumren har utelämnats här för att inte förvirra dig när du skriver in dem i ditt program.

#1: SKALOR

```
POKE 36878,15
FOR L = 250 TO 200 STEP -2
POKE 36876,L
FOR M = 1 TO 100
NEXT M
NEXT L
FOR L = 205 TO 250 STEP 2
POKE 36876,L
FOR M = 1 TO 100
NEXT M
NEXT L
POKE 36876,0
POKE 36878,0
```

#2: DATOR MUSIC

```
POKE 36878,15
FOR L = 1 TO 100
POKE 36876,INT(RND(1)*128) + 128
FOR M = 1 TO 10
NEXT M
NEXT L
POKE 36876,0
POKE 36878,0
```

#3: EXPLOSION

```
POKE 36877,220
FOR L = 15 TO 0 STEP -1
POKE 36878,L
FOR M = 1 TO 300
NEXT M
NEXT L
POKE 36877,0
POKE 36878,0
```

#4: FALLANDE BOMBER

```
POKE 36878,10
FOR L = 230 TO 128 STEP -1
POKE 36876,L
FOR M = 1 TO 20
NEXT M
NEXT L
POKE 36876,0
POKE 36877,200
FOR L = 15 TO 0 STEP -.05
POKE 36878,L
NEXT L
POKE 36877,0
```

#5: SIREN

```
POKE 36878,15
FOR L = 1 TO 10
FOR M = 180 TO 235 STEP 2
POKE 36876,M
FOR N = 1 TO 10
NEXT N
NEXT M
POKE 36876,0
FOR M = 1 TO 100
NEXT M
NEXT L
POKE 36878,0
```

#6: LASER STRÅLE

```
POKE 36878,15
FOR L = 1 TO 30
FOR M = 250 TO 240 STEP -1
POKE 36876,M
NEXT M
FOR M = 240 TO 250
POKE 36876,M
NEXT M
POKE 36876,0
NEXT L
POKE 36878,0
```

#7: HÖG-LÅG SIREN

```
POKE 36878,15
FOR L = 1 TO 10
POKE 36875,200
FOR M = 1 TO 500
NEXT M
POKE 36875,0
POKE 36876,200
FOR M = 1 TO 500
NEXT M
POKE 36876,0
NEXT L
POKE 36878,0
```

#8: UPPTAGET TON

```
POKE 36878,15
FOR L = 1 TO 15
POKE 36876,160
FOR M = 1 TO 400
NEXT M
POKE 36876,0
FOR M = 1 TO 400
NEXT M
NEXT L
POKE 36878,0
```

#9: RINGANDE TELEFON

```
POKE 36878,15
FOR L = 1 TO 5
FOR M = 1 TO 50
POKE 36876,220
FOR N = 1 TO 5
NEXT N
POKE 36876,0
NEXT M
FOR M = 1 TO 3000
NEXT M
NEXT L
POKE 36878,0
```

#10: KVITTRANDE FÅGLAR

```
POKE 36878,15
FOR L = 1 TO 20
FOR M = 254 TO 240 + INT (RND(1)*10) STEP -1
POKE 36876,M
NEXT M
POKE 36876,0
FOR M = 0 TO INT(RND(1)*100) + 120
NEXT M
NEXT L
```

#11: VIND

```
POKE 36878,15
POKE 36874,170
POKE 36877,240
FOR L = 1 TO 2000
NEXT L
POKE 36874,0
POKE 36877,0
POKE 36878,0
```

#12: VÅGOR

```
POKE 36877,180
FOR L = 1 TO 10
D = INT(RND(1)*5)*50 + 50
FOR M = 3 TO 15
POKE 36878,M
FOR N = 1 TO D
NEXT N
NEXT M
FOR M = 15 TO 3 STEP -1
POKE 36878,M
FOR N = 1 TO D
NEXT N
NEXT M
NEXT L
POKE 36878,0
POKE 36877,0
```

#13: FÖRSVINNANDE UFO

```
POKE 36878,15
FOR L = 130 TO 254
POKE 36876,L
FOR M = 1 TO 40
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

#14: LANDANDE UFO

```
POKE 36878,15
FOR L = 1 TO 20
FOR M = 220-L TO 160-L STEP -4
POKE 36876,M
NEXT M
FOR M = 160-L TO 220-L STEP 4
POKE 36876,M
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

#15: SKJUTANDE UFO

```
POKE 36878,15
FOR L = 1 TO 15
FOR M = 200 TO 220 + L*2
POKE 36876,M
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

#16: YLANDE VARG

```
POKE 36878,15
FOR L = 148 TO 220 STEP .7
POKE 36876,L
NEXT L
FOR L = 128 TO 200
POKE 36876,L
NEXT L
FOR L = 200 TO 128 STEP – 1
POKE 36876,L
NEXT L
POKE 36878,0
POKE 36876,0
```

#17: SPRINGANDE STEG

```
POKE 36878,15
FOR L = 1 TO 10
POKE 36874,200
FOR M = 1 TO 10
NEXT M
POKE 36874,0
FOR M = 1 TO 100
NEXT M
NEXT L
POKE 36878,0
```

#18: TICK-TACK

```
POKE 36878,15
FOR L = 1 TO 10
POKE 36875,200
FOR M = 1 TO 10
NEXT M
POKE 36875,0
FOR M = 1 TO 300
NEXT M
POKE 36874,200
FOR M = 1 TO 10
NEXT M
POKE 36874,0
FOR M = 1 TO 300
NEXT M
NEXT L
POKE 36878,0
```

#19: DÖRR SOM ÖPPNAS

```
POKE 36878,15
B = 0
FORL = 128 TO 255 STEP 11
POKE 36874,L
FOR M = 1 TO 10
NEXT M
B = B + 1
IF B = 3 THEN B = 0: POKE 36874,0
NEXT L
POKE 36874,0
POKE 36878,0
```

#20: BLIPPAR

```
POKE 36878,15
POKE 36876,220
FOR L = 1 TO 5
NEXT L
POKE 36876,0
FOR L = 1 TO 500
NEXT L
POKE 36876,200
FOR L = 1 TO 5
NEXT L
POKE 36876,0
FOR L = 1 TO 500
NEXT L
POKE 36878,0
```

BILAGA H: SKÄRMKODER

Följande tabell visar alla tecken som finns inbyggda i VIC 20. Tabellen visar vilket nummer du skall lägga in (POKE) i skärmminnet för att få det önskade tecknet. Den visar också vilket värde tecknet får om du PEEKar (läser) av minnespositionen.

Det finns två uppsättningar med tecken, men du kan bara använda ett i taget. Med detta menas att tecken från den ena uppsättningen inte kan visas samtidigt som tecken från den andra. Du byter teckenuppsättning genom att hålla SHIFT nedtryckt och trycka på COMMODORE-tangenten. Denna tryckning ändrar på den andra biten i minnespositionen 36869. Detta betyder att kommandot POKE 36869,240 gör att det blir stora bokstäver och grafik, POKE 36869,242 byter till stora och små bokstäver.

Om du vill göra avancerad animering så upptäcker du snart att det går bäst att använda kommandot POKE i stället för PRINT. Med POKE så lägger du in dina tecken och figurer i skärmminnet direkt. Vill du radera tecken så lägger du in koden 32, vilken motsvarar ett blanktecken.

Om du vill lägga in ett hjärta i positionen 7800, ta först reda på koden för ett hjärta i tabellen... koden för ett hjärta är 83... och skriv sedan POKE-kommandot med numret på positionen och numret på symbolen. Så här:

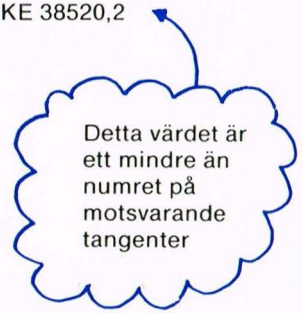
```
POKE 7800,83
```

Ett vitt hjärta kommer upp mitt på skärmen. Notera att tecknet kommer att vara osynligt om bakgrunden är vit. Försök att ändra positionen (se skärmkartan) eller skriv ut en annan symbol från tabellen.

Om du vill ändra färgen på symbolen studera då motsvarande karta för färgpositioner. Med andra ord det krävs ännu ett POKE-kommando för att få färg på tecknet.

För att t ex få ett rött hjärta, skriv följande:

```
POKE 38520,2
```



Detta ändrar färgen på symbolen i position 7800 till röd. Hade du haft en annan symbol där så hade även den blivit röd. Du kan med dessa två kartor skriva ut vilket tecken i vilken färg du vill på skärmen. Dessa POKE-kommandon kan läggas in i dina program och är mycket effektiva när du skall animera olika saker. PEEK är också användbart för att läsa vissa skärmpositioner när du gör avancerade program typ "STUDSANDE BOLL".

SKÄRMKODER

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
@		0	U	u	21	*		42
A	a	1	V	v	22	+		43
B	b	2	W	w	23	,		44
C	c	3	X	x	24	—		45
D	d	4	Y	y	25	.		46
E	e	5	Z	z	26	/		47
F	f	6	[(Ä)	ä	27	ø		48
G	g	7	£(Ö)	ö	28	1		49
H	h	8	](Å)	å	29	2		50
I	i	9	↑		30	3		51
J	j	10	←		31	4		52
K	k	11	SPACE		32	5		53
L	l	12	!		33	6		54
M	m	13	"		34	7		55
N	n	14	#		35	8		56
O	o	15	$		36	9		57
P	p	16	%		37	:		58
Q	q	17	&		38	;		59
R	r	18	'		39	<		60
S	s	19	(		40	=		61
T	t	20	)		41	>		62

SET 1	SET 2	POKE
?		63
		64
	A	65
	B	66
	C	67
	D	68
	E	69
	F	70
	G	71
	H	72
	I	73
	J	74
	K	75
	L	76
	M	77
	N	78
	O	79
	P	80
	Q	81
	R	82
	S	83

SET 1	SET 2	POKE
	T	84
	U	85
	V	86
	W	87
	X	88
	Y	89
	Z	90
	Ä	91
	Ö	92
	Å	93
		94
		95
SPACE		96
		97
		98
		99
		100
		101
		102
		103
		104
		105

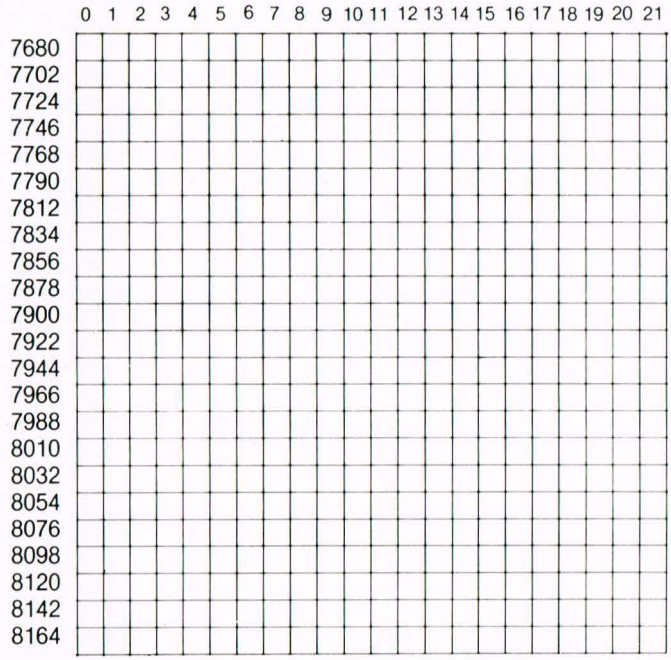
SET 1	SET 2	POKE
		106
		107
		108
		109
		110
		111
		112
		113
		114
		115
		116
		117
		118
		119
		120
		121
	✓	122
		123
		124
		125
		126
		127

BILAGA I: SKÄRMKARTOR

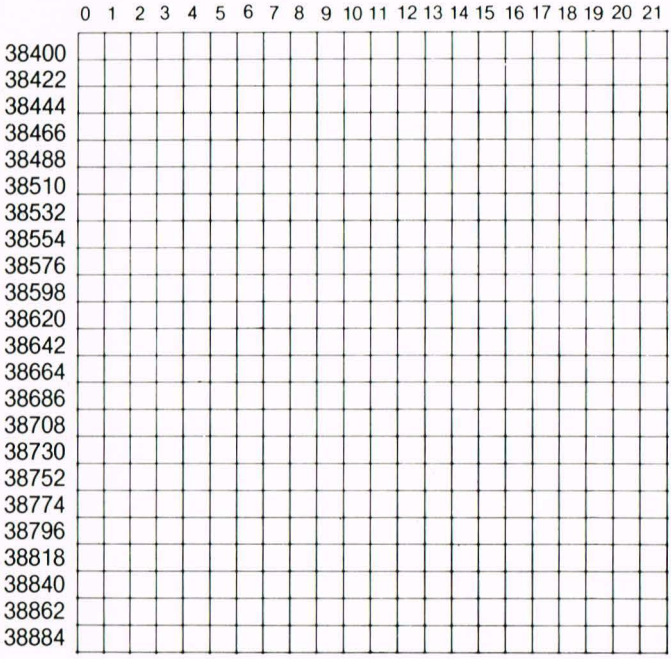
Använd dess kartor för att ta reda på en position i skärmen. Leta rätt på din ruta i kartan och addera sedan numren i raden och kolumnen. Om du till exempel vill skriva ut en boll mitt på skärmen adderar du numren på rad 11 och kolumn 11 (7900+10) vilket blir 7910. Om du lägger in koden för en boll (81, se bilaga H) i position 7910 genom att skriva POKE 7910,81 så kommer det fram en vit boll på skärmen. För att ändra bollens eller något annat teckens färg, så letar du upp motsvarande position på färgkartan och adderar numret på raden och kolumnen (38620+10 = 38630) och skriver ett andra POKE-kommando. Ändrar du färgkoden i denna position genom att skriva POKE 38630,5 så ändrar bollen sin färg till grön. Notera att värdet på färgkoden är ett steg lägre än motsvarande tangenter. Se tabellen.

Lista över färgkoder:

Kod	Färg
0	Svart
1	Vit
2	Röd
3	Cyan
4	Violett
5	Grön
6	Blå
7	Gul



SIDA 1: SKÄRMTECKEN KODER



SIDA 2: FÄRGKODER MINNESKARTA

BILAGA J: ASCII OCH CHR$-KODER

Denna tabell visar vad som skrivs ut när du ger kommandot PRINT CHRS(X), där X är alla värden mellan 0 och 255. Den visar också vilket värde du får om du skriver PRINT ASC("X") där X är vilket tecken som helst. Detta är användbart när du skall analysera tecken från ett GET-kommando, eller konvertera stora/små bokstäver.

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
	0		16	SPACE	32	ø	48
	1	CRSR ↓	17	!	33	1	49
	2	RVS ON	18	"	34	2	50
	3	CLR HOME	19	#	35	3	51
	4	INST DEL	20	$	36	4	52
VIT	5		21	%	37	5	53
	6		22	&	38	6	54
	7		23	'	39	7	55
	8		24	(	40	8	56
	9		25	)	41	9	57
	10		26	*	42	:	58
	11		27	+	43	;	59
	12	RÖD	28	,	44	<	60
RETURN	13	CRSR →	29	—	45	=	61
BYT TILL GEMENER	14	GRÖN	30	.	46	>	62
	15	BLÅ	31	/	47	?	63

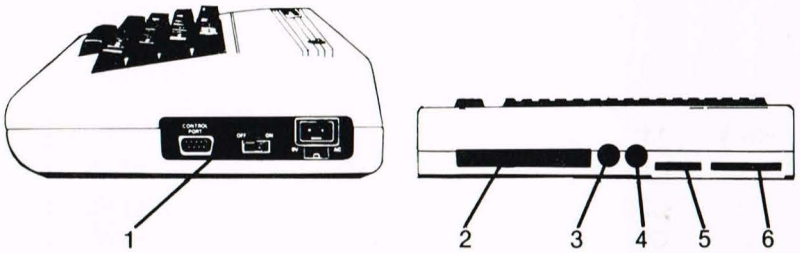
PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
@	64	U	85		106		127
A	65	V	86		107		128
B	66	W	87		108		129
C	67	X	88		109		130
D	68	Y	89		110		131
E	69	Z	90		111		132
F	70	[(Ä)	91		112	f1	133
G	71	∉ (Ö)	92		113	f3	134
H	72	] (Å)	93		114	f5	135
I	73	↑	94		115	f7	136
J	74	←	95		111	f2	137
K	75		96		117	f4	138
I	76		97		118	f6	139
M	77		98		119	f8	140
N	78		99		120	SHIFT RETURN	141
O	79		100		121	BYT TILL VERSALER	142
P	80		101		122		143
Q	81		102		123	SVART	144
R	82		103		124	CRSR	145
S	83		104		125	RVS OFF	146
T	84		105		126	CLR HOME	147

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
INST DEL	148	CYN	159		170		181
	149	SPACE	160		171		182
	150		161		172		183
	151		162		173		184
	152		163		174		185
	153		164		175		186
	154		165		176		187
	155		166		177		188
VIOL.	156		167		178		189
CRSR	157		168		179		190
GUL	158		169		180		191

BILAGA K:
MATEMATISKA FUNKTIONER

FUNKTION	VIC BASIC MOTSVARANDE
SECANT	SEC(X) = 1/COS(X)
COSECANT	CSC(X) = 1/SIN(X)
COTANGENT	COT(X) = 1/TAN(X)
INVERSE SINE	ARCSIN(X) = ATN(X/SQR(– X*X + 1))
INVERSE COSINE	ARCCOS(X) = – ATN(X/SQR (– X*X + 1)) + π/2
INVERSE SECANT	ARCSEC(X) = ATN(X/SQR(X*X – 1))
INVERSE COSECANT	ARCCSC(X) = ATN(X/SQR(X*X – 1)) + (SGN(X) – 1)* π/2
INVERSE COTANGENT	ARCOT(X) = ATN(X) + π/2
HYPERBOLIC SINE	SINE(X) = (EXP(X) – EXP(– X))/2
HYPERBOLIC COSINE	COSH(X) = (EXP(X) + EXP(– X))/2
HYPERBOLIC TANGENT	TANH(X) = EXP(– X)/EXP(X) + EXP (– X))*2 + 1
HYPERBOLIC SECANT	SECH(X) = 2/(EXP(X) + EXP(– X)
HYPERBOLIC COSECANT	CSCH(X) = 2/(EXP(X) – EXP(– X))
HYPERBOLIC COTANGENT	COTH(X) = EXP(– X)/(EXP(X) – EXP(– X)*2 + 1
INVERSE HYPERBOLIC SINE	ARCSINH(X) = LOG(X + SQR(X*X + 1)
INVERSE HYPERBOLIC COSINE	ARCCOSH(X) = LOG(X + SQR(X*X – 1)
INVERSE HYPERBOLIC TANGENT	ARCTANH(X) = LOG((1 + X)/(1 – X))/2
INVERSE HYPERBOLIC SECANT	ARCSECH(X) = LOG((SQR (– X*X + 1) + 1)/X)
INVERSE HYPERBOLIC COSECANT	ARCCSCH(X) = LOG((SGN(X)*SQR (X*X + 1)/X
INVERSE HYPERBOLIC COTANGENT	ARCCOTH(X) = LOG((X + 1)/(X – 1))/2

BILAGA L:
KONTAKTPLACERING FÖR I/O



1) Spel I/O

2) Minnesexpansion

3) Ljud och Video

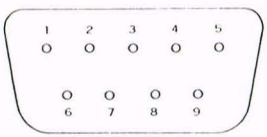
4) Serial I/O (flexskivminne, printer)

5) Kassett

6) Användar port (modem)

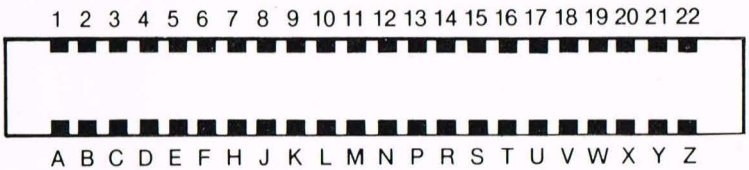
---

1) SPEL I/O



PIN #	TYP	ANM.
1	JOYØ	
2	JOY1	
3	JOY2	
4	JOY3	
5	POT Y	
6	LJUSPENNA	
7	+5V	MAX. 1ØØmA
8	GND	
9	POT X	

2) MINNESEXPANSION



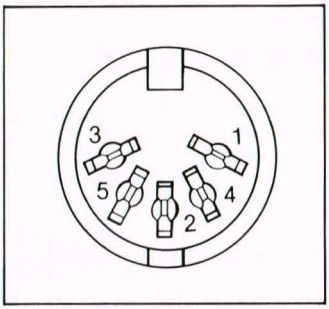
PIN #	TYP
1	GND
2	CDØ
3	CD1
4	CD2
5	CD3
6	CD4
7	CD5
8	CD6
9	CD7
1Ø	$\overline{\text{BLK1}}$
11	$\overline{\text{BLK2}}$

PIN #	TYP
12	$\overline{\text{BLK3}}$
13	$\overline{\text{BLK5}}$
14	$\overline{\text{RAM1}}$
15	$\overline{\text{RAM2}}$
16	$\overline{\text{RAM3}}$
17	VR/W
18	CR/W
19	$\overline{\text{IRQ}}$
2Ø	NC
21	+5V
22	GND

PIN #	TYP
A	GND
B	CAØ
C	CA1
D	CA2
E	CA3
F	CA4
H	CA5
J	CA6
K	CA7
L	CA8
M	CA9

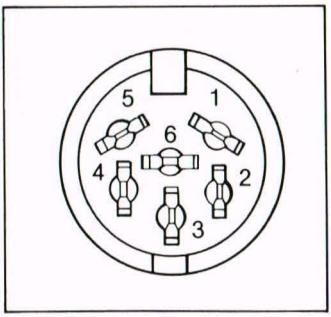
PIN #	TYP
N	CA1Ø
P	CA11
R	CA12
S	CA13
T	I/Ø2
U	I/Ø3
V	SØ2
W	$\overline{\text{NMI}}$
X	$\overline{\text{RESET}}$
Y	NC
Z	GND

3) LJUD/VIDEO



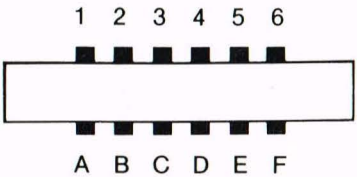
PIN #	TYP	ANM.
1	+6V	10mA MAX
2	GND	
3	LJUD	
4	VIDEO LÅG	
5	VIDEO HÖG	

4) SERIAL I/O



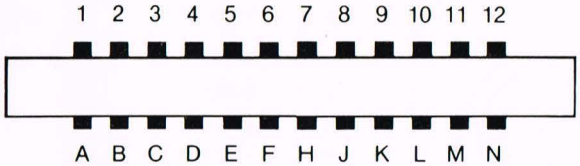
PIN #	TYP
1	SERIAL SRQ IN
2	GND
3	SERIAL ATN IN/OUT
4	SERIAL CLK IN/OUT
5	SERIAL DATA IN/OUT
6	NC

5) KASSETT



PIN #	TYP
A-1	GND
B-2	+5V
C-3	KASSETT MOTOR
D-4	KASSETT LÄS
E-5	KASSETT SKRIV
F-6	KASSETT SWITCH

6) ANVÄNDAR PORT



PIN #	TYP	ANM.	PIN #	TYP	ANM.
1	GND		A	GND	
2	+5V	1ØØmA MAX.	B	CB1	
3	$\overline{\text{RESET}}$		C	PBØ	
4	JOYØ		D	PB1	
5	JOY1		E	PB2	
6	JOY2		F	PB3	
7	LJUSPENNA		H	PB4	
8	KASSETT SWITCH		J	PB5	
9	SERIAL ATN IN		K	PB6	
1Ø	+9V	1ØØmA MAX.	L	PB7	
11	GND		M	CB2	
12	GND		N	GND	

BILAGA M: VIC PROGRAM ATT SKRIVA

```
1 REM TANK-V-UFO BY D.LATER
5 OF=30720
6 VA=9*16↑3+14:VN=9*16↑3+13
8 POKEVA+1,25
9 PRINT"♥QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ";
20 PRINT"S** TANK VERSUS UFO **"
21 PRINT"Z-LEFT,C-RIGHT,B-FIRE"
100 PRINT"SQQQQQQQQQQQQQQQQQQQQQQQ"
101 FORA=38400TO38400+22*23   :POKEA,0:NEXT:A=0
105 PRINT"π    |"
110 PRINT" _▚  ■_"
115 PRINT" ▶OOOO| "
120 PRINT"▐▐▐";
125 IFPEEK(197)=34THENGOSUB300:A=A+1:GOTO220
130 IFPEEK(197)=33THENGOSUB300:A=A-1:GOTO220
135 IFPEEK(197)=35ANDB=0THENB=1:C=8079+A:L=1:P=15:POKEVN,158:POKEC,160:POKEC+OF
.4:GOTO170
136 IFL=0THEN140
137 POKEVA,P:P=P-1:IFP=-1THENL=0:POKEVN,0
140 IFB=0THEN170
141 FORH=1TO2
145 POKEC,32:C=C-22
150 IFPEEK(C)=60ORPEEK(C)=42ORPEEK(C)=62    THEN600                    :
155 IFC<7746THENB=0:GOTO170          :
160 POKEC,46
161 NEXT
170 IFD=0THEN800
172 IFD=0THEN180
```

NYCKEL TILL SPECIAL SYMBOLER

```
"♥" IS CLR
"S" IS HOME
"Q" IS CRSR DOWN
"▐" IS CRSR LEFT
"□" IS CRSR UP
"@" IS CTRL-2
"▚" IS CTRL-3
"▒" IS CTRL-5
"■" IS CTRL-1
```

```
174 POKEE,32:POKEE-1,32:POKEE-2,32:K=K+1
176 IFE=ITHEND=0:GOTO180
178 E=E+J: POKEE,62:POKEE-1,42:POKEE-2,60
179 IFJ=1THEN182
180 IFINT((8186-E)/22)=22-K-AANDF=0THEN F=1:G=E+21:M=21:GOTO183       :
181 GOTO183
182 IFINT((8098-E)/22)=A-KANDF=0THENF=1:G=E+23:M=23
183 IFF=0THEN125
184 POKEG,32:G=G+M
186 IFPEEK(G)<>32THEN700
187 IFG>7680+22*21THENF=0:GOTO500
189 POKEG,81:GOTO125
220 IFA<0THENA=0
221 IFA>15THENA=15
222 PRINTTAB(A)"    |"
225 PRINTTAB(A)" _■  ■_ "
230 PRINTTAB(A)" |oooo| "
235 PRINT"▒▒▒";:GOTO135
300 PRINTTAB(A)"      "
305 PRINTTAB(A)"      "
310 PRINTTAB(A)"       "
315 PRINT"▒▒▒";:RETURN
400 PRINTPEEK(197):GOTO400
500 POKEG,66:POKEG+1,78:POKEG-1,77:POKEG-20,46:POKEG-21,46:POKEG-22,46
510 POKEG-23,46:POKEG-24,46
520 FORAA=1TO100:NEXT
530 POKEG,32:POKEG+1,32:POKEG-1,32:POKEG-20,32:POKEG-21,32:POKEG-22,32
535 POKEG-23,32:POKEG-24,32
590 GOTO125
600 POKEC,160:POKEC+1,160:POKEC-1,160:POKEC+22,160:POKEC-22,160
601 L=0
```

```
610 POKEVN,128+100
611 FORGG=15TO0STEP-1:POKEVA,GG:FORGH=1TO70:NEXT:NEXT
615 B=0:D=0
616 POKEC,32 :POKEC+1, 32:POKEC-1, 32:POKEC+2, 32:POKEC-2,32:POKEC+3,32:POKEC-3
,32
617 POKEC-22,32:POKEC+22,32
640 E=E+22+J:POKEVA,15:POKEVN,0:
645 POKEE,62:POKEE-1,42:POKEE-2,60
646 FORO=248TO253:POKEVN-1,O:NEXT:FORO=253TO248STEP-1:POKEVN-1,O:NEXT
647 POKEE,32:POKEE-1,32:POKEE-2,32
650 IFE<7680+20*22THEN640
651 E=E+J
652 POKEE+22,62:POKEE+21,42:POKEE+20,60: POKEE+OF,4:POKEE+OF-1,4:POKEE+OF-2,4
653 POKEE+22+OF,0:POKEE+21+OF,0:POKEE+20+OF,32
654 POKEVN-1,0:POKEVN,128:FORQ=1TO20:POKEVA,15-INT(Q/1.33)
655 POKEE,223:POKEE-1,223:POKEE-2,223:FORO=1TO80:NEXT
656 POKEE,233:POKEE-1,233:POKEE-2,233:FORO=1TO80:NEXT
657 NEXT:POKEE,32:POKEE-1,32:POKEE-2,32:POKEE+22,32:POKEE+21,32:POKEE+20,32 :
658 PRINT"■■■■                    "
659 DU=DU+1:PRINT"■■■■UFOS"DU"TANKS"DT:PRINT"■■■■■■■■■■■■■■■■■■■■"
660 GOTO125
700 POKEVN,128:L=0
701 A=A+1: FORKL=1TO200:POKEVA,15-INT(KL/13):
704 PRINTTAB(A)"■■■■■"
705 PRINTTAB(A)"■■    "
715 PRINT"■■";
720 PRINTTAB(A)"■■■■■"
725 PRINTTAB(A)"■■    "
735 PRINT"■■";
740 NEXT
745 PRINTTAB(A)"    "
```

```
 750 PRINTTAB(A)"     "
 751 PRINTTAB(A-1)"        ";
 752 PRINT"   ";
 756 PRINT"          ;
 752 PRINT"   ";
 756 PRINT"                          "
 757 DT=DT+1:PRINT"     JFOS"DU"TANKS"DT:PRINT"                    "
 760 F=0:  A=0:GOTO105
 800 D=1:E=7702+INT(RND(1)*14)*22+88 :I=E-20:K=0:J=-1:IFRND(1)>.5THENE=E-21:I=E+
20:J=1
 805 GOTO172
```

```
  0 REM"       "   KILLER COMET        BY  DUANE LATER
  1 C=7680+22*6:DIMB(12):LL=C:POKE9*16↑3+15,8
  2 PRINT"  *** KILLER COMET  ***":T=0
  3 PRINT"    HIT ANY KEY"
  8 POKE8179,160:
  9 FORA=38444TO38400+505:POKEA,1:NEXT
  10 :
  20 FORA=1 TO 12:B(A)=160:NEXT
  21 IFW=0THEN25
  22 POKE9*16↑3+14,U:U=U-2:IFU<=0THENW=0:POKE9*16↑3+13,0
  25 F=1:REM ERA METEOR
  26 FORE=0TO44STEP22
  27 FORD=C+ETOC+3+E:POKED,32 :F=F+1:NEXT:NEXT
  28 C=C+1
  29 F=1:REM DRAW METEOR
  30 POKE8179,160:FORE=0TO44STEP22
  35 FORD=C+ETOC+3+E:POKED,B(F):F=F+1:NEXT:NEXT
```

```
 36 IFPEEK(8178)=160THENPRINT"□□□□□MOON BASE DESTROYED !":GOTO500
 40 GETA$:IFA$<>""ANDG=0THENG=1:S=7680+15+22*21
 50 IFG=0THEN80
 55 POKES,32:S=S-22
 60 IFS<7746THENG=0:GOTO21
 70 IFPEEK(S)=160THENPOKES,32:G=0:T=T+1:W=1:POKE9*16↑3+13,128+000:U=15:GOTO80
 71 IFPEEK(S-1)=160THENG=0:POKES-1,32:T=T+1:W=1:POKE9*16↑3+13,128+000:U=15:GOTO8
0
 75 POKES,81
 80 F=1:REM CHECK MET
 81 IFT=12THENPRINT"□□□□***METEOR DESTROYED***":FORRR=1TO2500:NEXT:LL=LL+44:C=LL
:GOTO2
 82 FORE=0TO44STEP22
 84 FORD=C+ETOC+3+E:IFPEEK(D)=32THENB(F)=32
 86 F=F+1:NEXT:NEXT
 90 GOTO21
 500 POKE9*16↑3+13,128+5
 505 POKE9*16↑3+14,5:FORRR=1TO300:NEXT
 510 FORA=15TO0STEP-1
 511 POKE9*16↑3+14,A
 520 FORRR=1TO500:NEXT
 530 NEXT
 540 FORRR=1TO2000:NEXT:RUN
READY.
```

```
 1 REM"███████"
 2 REM"████████"    ROCKET COMMAND        BY  DUANE LATER
 10 VI=9*16↑3:OF=38400-7680:PRINT"]"
 11 FORA=38400+22TO38400+22*23:POKEA,0:NEXT
 15 C=7680+22*20+15
 20 POKEVI+15,6+128+64+32+8
 30 PRINT"S\*** ROCKET COMMAND ***";
 35 PRINT"      HIT ANY KEY"
 40 PRINT"QQQQQQQQQQQQQQQ█×▌Q×▌Q×▌Q×▌Q×▌Q×▌Qooooooooooooooo×";
 50 PRINT"]███████████████↑↑↑↑↑↑↑↑↑↑↑↑↑↑"
 60 A=7680+22*22+15
 70 GET A$:IFA$<>""ANDB=0THENB=1:POKEC+22,32:D=C:C=C-1:K=1:POKE9*16↑3+13,128+125
 L=16
 71 IFK=1THENL=L-1:POKE9*16↑3+14,L
 72 IFL=0THENK=0:POKE9*16↑3+13,0
 75 IFC=8121THENPRINT"SQQQQQQNO MORE ROCKETS!!!":POKE9*16↑3+14,0: FORAA=1TO3000:
NEXT:RUN
 80 IFB=0THEN110
 85 POKED,32:D=D-22
 90 IFPEEK(D)=60ORPEEK(D)=62THENGOSUB500:GOTO70
 100 IFD<7680+88THENB=0:GOTO110
 105 POKED,30
 110 IFH=0THEN200
 115 IFH=0THEN70
 120 POKEF,32:F=F+E
 125 IFF=ITHENH=0:GOTO70
 130 IFPEEK(F)=30THENGOSUB500:GOTO70
 140 POKEF,G:GOTO70
 200 H=1
 205 IFINT(RND(1)*2)=1THENE=-1:F=7702+(INT(RND(1)*10)+6)*22:I=F-22:G=60:   GOTO11
5
```

```
 210 E=1:F=7680+(INT(RND(1)*10)+6)*22:I=F+22:G=62: GOTO115
 500 B=0:H=0
 501 SC=SC+10:PRINT"■■■■■SCORE = "SC
 502 POKEF+OF,4:POKEF+1+OF,4:POKEF-1+OF,4
 503 POKEF+OF+22,4:POKEF-22+OF,4
 510 POKEF,160:POKEF+1,160:POKEF-1,160:POKEF+22,160:POKEF-22,160
 521 POKE9*16↑3+13,128+35
 522 FORY=16TO0STEP-1
 523 POKE9*16↑3+14,Y
 524 FORP=1TO80:NEXT:NEXT
 530 POKEF,32:POKEF+1,32:POKEF-1,32:POKEF+22,32:POKEF-22,32
 531 POKEF+OF,0:POKEF+1+OF,0:POKEF-1+OF,0
 532 POKEF+OF+22,0:POKEF-22+OF,0
 533 POKE9*16↑3+13,0
 540 FORGH=FTOF+22*16STEP22
 544 II=PEEK(GH):POKEGH,G:FOROO=1TO60:NEXT
 546 POKEGH,II:NEXT
 800 RETURN
 999 GOTO70
 1000 POKE9*16↑3+13,128+125
 1001 FORY=16TO0STEP-1
 1005 POKE9*16↑3+14,Y
 1010 NEXT:POKE9*16↑3+13,0
 1020 RETURN
READY.
```

BILAGA N: FELMEDDELANDEN

Denna bilaga innehåller en komplett lista på felmeddelanden som ses av VIC 20, med en beskrivning av orsaken.

BAD DATA... Text data lästes in av en fil, men programmet väntade sig numeriska data.

BAD SUBSCRIPT... Programmet försökte referera till ett element i en matris som ligger utanför området i DIM-kommandot.

CAN'T CONTINUE...CONT-kommandot fungerar inte om, programmet aldrig körts, det har blivit ett fel eller en rad har blivit redigerad.

DEVICE NOT PRESENT... Den begärda in/ut enheten fanns inte tillgänglig för ett OPEN, CLOSE, CMD, PRINT , INPUT , eller GET .

DIVISION BY ZERO... Division med noll är en matematisk orimlighet och tillåts inte.

EXTRA IGNORED...För många svar har givits in på ett INPUT-kommando. Endast de första svaren räknas.

FILE NOT FOUND... Filen som specifiserades i ett CLOSE, CMD, PRINT , INPUT eller GET , måste först öppnas.

FILE OPEN... Ett försök att öppna en fil med ett nummer som redan använts av en annan fil.

FORMULA TO COMPLEX... Uttrycket bör delas upp i minst två separata delar.

ILLEGAL DIRECT... Input-kommandot kan endast användas i program och inte i direkt mode.

ILLEGAL QANTITY... Ett värde som används som parameter i en funktion ligger utanför det tillåtna området.

LOAD... Det är problem med läsning av programmet från bandet.

NEXT WITHOUT FOR... Detta orsakas av oriktigt nästade loopar eller att variabeln i NEXT-kommandot inte stämmer överens med den i FOR-kommandot.

NOT INPUT FILE... Ett försök att göra INPUT eller GET till en fil som öppnats för enbart skrivning.

NOT OUTPUT FILE... Ett försök har gjort att använda PRINT till en fil som öppnats enbart för läsning.

OUT OF DATA... Ett READ-kommando utförs utan att det finns data kvar som är oläst.

OUT OF MEMORY... Det finns inte plats för mera program eller variabler i RAM-minnet. Detta kan också uppstå när för många nästade FOR... NEXT loopar används eller för många GOSUB nivåer finns.

OVERFLOW... Resultatet av en operaton är större än det största tal som tillåts, vilket är 1.70141884E+38.

REDIM'D ARRAY... En matris kan bara dimensioneras en gång. Om en matris-variabel används innan den dimensionerats sker en automat-dimensionering och sätter antalet element till 10 och ett efterföljande DIM-kommando orsakar detta fel.

REDO FROM START... Text skrevs in i ett INPUT-kommando där det förväntades ett numeriskt värde. Skriv bara om ditt svar så fortsätter programmet av sig själv.

RETURN WITHOUT GOSUB... Ett RETURN-kommandot utfördes utan ett föregående GOSUB.

STRING TOO LONG... En sträng får inte vara längre än 255 tecken.

SYNTAX... Ett kommando är givet som inte VIC kan förstå. En missad parentes eller ett felstavat kommando.

TYPE MISMATCH... Detta förekommer när en sträng skrivs istället för en numerisk variabel eller tvärtom.

UNDEF'D FUNCTION... En användardefinierad funktion har använt utan att den tidigare har definierats.

UNDEF'D STATMENT... Ett försök att göra ett GOTO, GOSUB eller ett RUN-kommando till en programrad som inte finns.

VERIFY... Programmet på band eller diskett stämmer inte överens med det i VIC's minne.

SAKREGISTER

A

Addition 24, 115
Adress etikett program 92
AND operation 115
Animering 50, 51-66, 99, 139, 143
Anslutning av VIC till TV-monitor V
Användardefinierade
  funktioner (DEF) 120
Aritmetiska operationer 24, 116
Aritmetiska uttryck 24, 115, 123, 148
Array 114-120
ASC funktionen 131, 145
ASCII tecken koder 145
ATN funktionen 128

B

BASIC
  förkortningar 133
  kommando 115
  operationer 115
  kommando 119
  variabler, 86, 113
Beräkningar 24
Brus 71, 74
Buffer 110

C

CHR$ funktionen 102, 130, 145
Citationstecken 96
CLR-kommandot 119
CLR/HOME-tangenten 6, 18
CLOSE-kommandot 110, 119
Commodore-tangent (se grafik-tangent)
CONT-kommandot 115
COSinus funktionen 129
CRSR-tangenter 18, 60
CTRL-tangenten 18

D

DATA-kommandot 79, 119
Data spara och hämta från tape 109
DEF-kommandot 119
DEL-tangenten 3, 8, 19
DIMensioneringskommando 120
Division 115, 160

E

Editering av program 8, 50
END-kommandot 121
Expansionsport 106, 149
Exponetieringsfunktion 129

F

Felmeddelande 6, 160
Filer kassettband 109
FOR-kommandot 121
FOR...NEXT loop 121
FRE funktionen 131
Funktioner 129
Färg
  tangenter 19, 32
  minneskarta 63, 143, 144
  bakgrunds och ram 33, 36, 38, 39, 134
Fördröjningsloop 55, 78, 96
Förkortningar BASIC-kommandot 133

G

Gemena bokstäver 20
GET-kommandot 89, 122
GET kommandot 123, 111
GOSUB-kommandot 122
GOTO-kommandot 123
Grafiska tangenter 14, 19
Grafiska symboler 14, 142, 146-147

H

Heltalsvariabler 113
Hyberboliska funktioner 148

I

IEEE-488 interface 107
IF...THEN kommandot 123
INPUT-kommandot 84, 124
INPUT kommandot 111
INST-tangenten 3, 8, 19
INT-funktionen 129
In/ut-kontakter 149
In/ut-portar 106, 149

J

Joysticks 108

K

Kassettport 106
Kassettbandspelare 107, 109
Kassettoperationer 109
Keyboard 17-20
Klocka 114
Kommando, BASIC 115

L

Ladda program från band 109
LEFTS funktionen 131
LEN-funktionen 131
LET-kommandot 124
Likamed tecken 115
LIST-kommandot 8, 9, 50, 116
Ljudeffekter 135
LOAD-kommandot 109, 116
LOG-funktionen 130
Loopar, tidfördröjning 55, 78, 96

M

Markör (crsr) 3, 18, 60
Matematiska
  formler 24, 115, 123, 148, 160
  funktionstabell 148
  symboler 115
MIDS funktionen 43, 131
Mindre än 115
Modulator RF VI
Multiplikation 115
Musik
  ton 68
  ljudeffekter 135
  tabell över noter 73
  VIC piano 75
  skriva melodier 77

N

Namn
  program 109
  variabler 86
NEW-kommandot 7
NEXT-kommandot 125
NOT-operation 115
Numeriska variabler 86, 113

O

ON-kommandot 125
OPEN-kommandot 126
Operationer
  aritmetiska 24, 114
  logiska 115
  relation 115

P

Parenteser 115
PEEK-funktionen 130
Periferienheter 106
Pi 20
POKE-kommandot 36, 80, 125
Portar in/ut 149
POS-funktionen 132
PRINT-kommandot 5, 21, 126
PRINT 127
Program
  editering 8, 50
  radnummer 79
  hämta/spara på band 109

R

Radnummer 78
READ-kommandot 79, 128
REM-kommandot 128
Reserverade ord 86
Reset (se Restore-tangent)
Restore-tangent 17, 26
RESTORE-kommando 128
Return-tangent 18
RETURN-kommandot 128
RIGHTS funktionen 132
RND-funktionen 40, 43, 130
RUN-kommando 117
RUN/STOP-tangent 19
Rätta skrivfel 8, 50

S

SAVE-kommandot 109, 117
Serie port 107
SGN-funktionen 130
Shift-tangenten 18
Siffror 13, 24, 115, 123
SINus funktionen 131
Skriva till band 109
Skärmminneskarta 63, 143-144
Slumptal 40, 43, 103
Spara program på band 109
Spel att försöka 153
Spelkontroll 108
Spel port 108, 149
SPC-funktion 132
SQR-funktionen 131
ST variabel
STOP-tangent 19
Sträng variabler 42, 86, 113
STRS funktion 132
Större än 115
Subskription 115
SYS-kommandot 128
Syntax error 6

T

Tab funktionen 132
TAN-funktionen 131
TI variabel 114
TIS variabel 114
Tid, ställa klockan 114
Tillbehör 106, 109
TO kommandot 121
  Toner 70
TV, uppkoppling till VIC V

U

USR-funktionen 131
Upphöjt till 115

V

VAL funktionen 132
Variabler
  matris 114
  flyttal 113
  heltal 113
  numeriska 86, 113
  sträng 42, 86, 113
Versaler 20
VERIFY kommandot 109, 118
VIC tips 3, 8, 16, 39, 40, 47, 50, 78, 96
Video port V, 50
Volym 69, 71, 72

W

WAIT kommandot 129

Anteckningar

EN ANVÄNDARVÄNLIG DATOR

Den nya VIC-datorn är konstruerad för att vara den mest användarvänliga folkdatorn på marknaden. Användarvänlig i pris och storlek och enkel att expandera.

Med VIC, hjälper COMMODORE dig att bli insatt i datorernas värld på ett enkelt och lättförståeligt sätt... med tillräckligt inbyggda expansionsmöjligheter för att låta systemet växa i takt med dina ökande kunskaper och därmed ökande krav på sofistikerade datorsystem.

VIC-ägare som vill veta mera om datorer skall fråga sin återförsäljare om detta material för självstudie:

VIC LÄROSERIE... ett bibliotek med självinstruerande böcker och kassettband/pluginkort som hjälper dig att förstå hur datorer och programmering fungerar.

VIC PROGRAMMERINGSHANDBOK... en kompakt bok om VIC 20 som beskriver och förklarar allt som kan vara nyttigt att veta av den som programmerar VIC. Lämplig för både nya och erfarna programmerare.

VIC-PROGRAM KASSETTER, MODULER OCH DISKETTER... ett växande bibliotek av utbildnings- och avkopplingsprogram som gör att du kan använda din VIC för att lösa problem, utveckla din inlärningsförmåga eller att spela intressanta TV-spel. Dessa enkla program kräver ingen tidigare erfarenhet av datorer.

commodore

handic
electronic ab

Box 1063, 436 00 Askim/Göteborg, Tel: 031/28 97 90
– ett företag i Datatronicgruppen–

Bredängs Repro AB 1981